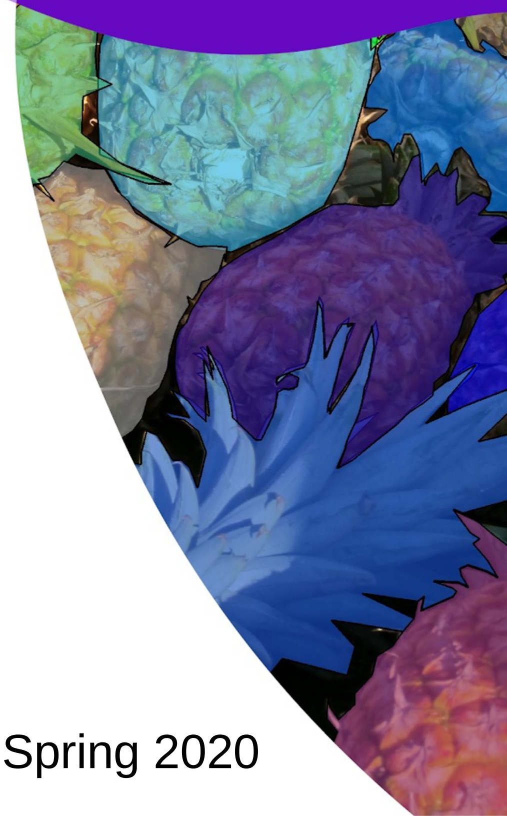
NEW YORK UNIVERSITY

Energy-Based Models (part 1)

http://bit.ly/DLSP20

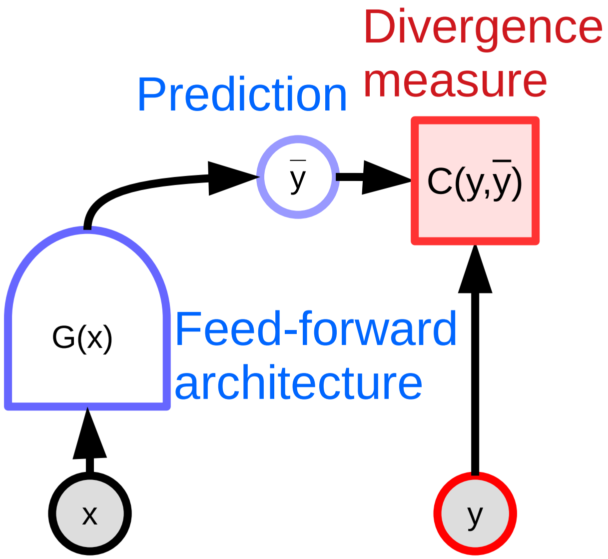
Yann LeCun
NYU - Courant Institute & Center for Data Science
Facebook AI Research
http://yann.lecun.com
TAs: Alfredo Canziani, Mark Goldstein



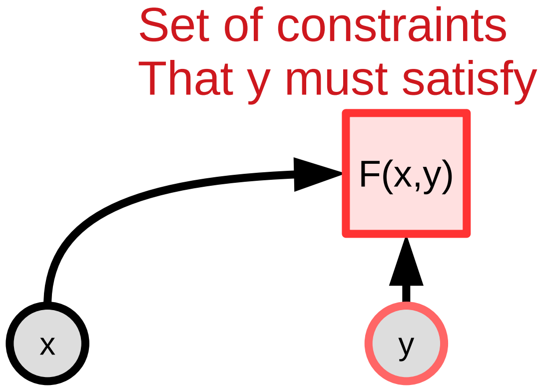
Deep Learning, NYU, Spring 2020

Energy-Based Models

- **Feed-forward nets use a finite number of steps to produce a single output.**
- **What if…**
  - The problem requires a complex computation to produce its output? (complex inference)
  - There are multiple possible outputs for a single input? (e.g. predicting future video frames)

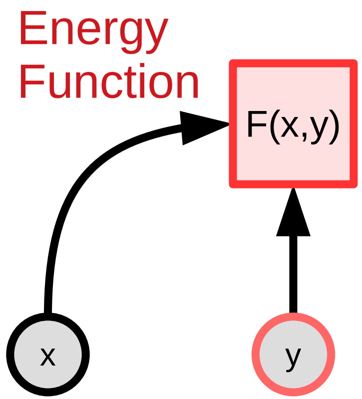


- **Inference through constraint satisfaction**
  - Finding an output that satisfies constraints: e.g a linguistically correct translation or speech transcription.
  - Maximum likelihood inference in graphical models



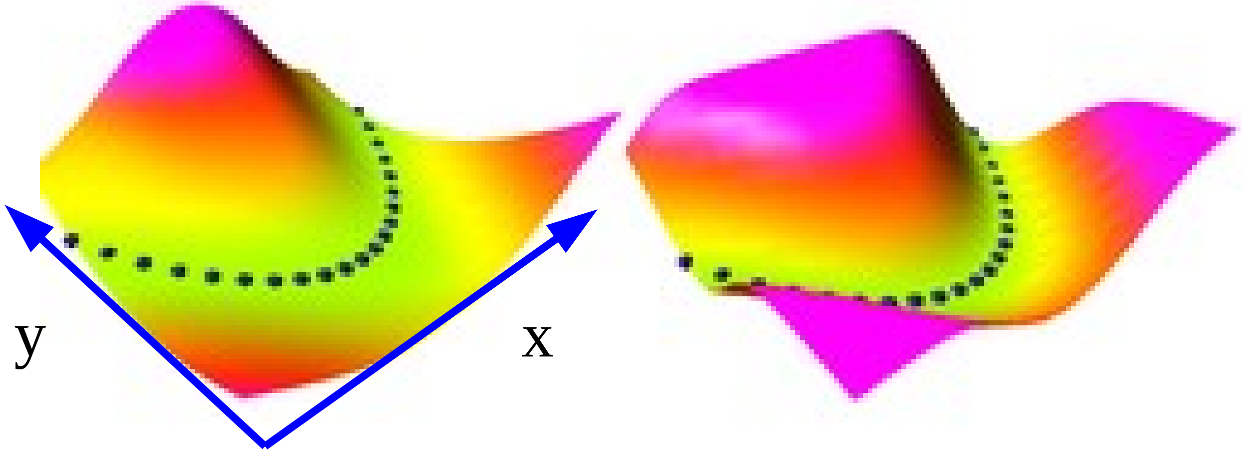
Energy-Based Models (EBM)

- **Energy function F(x,y) scalar-valued.**
  - Takes low values when y is compatible with x and higher values when y is less compatible with x
- **Inference: find values of y that make F(x,y) small.**
  - There may be multiple solutions $\check{y} = \text{argmin}_y F(x, y)$



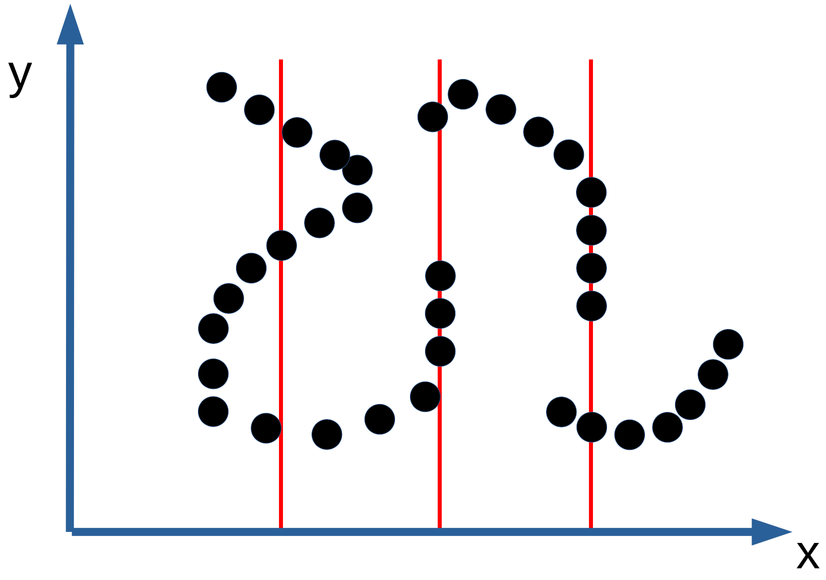
- **Note: the energy is used for inference, not for learning**

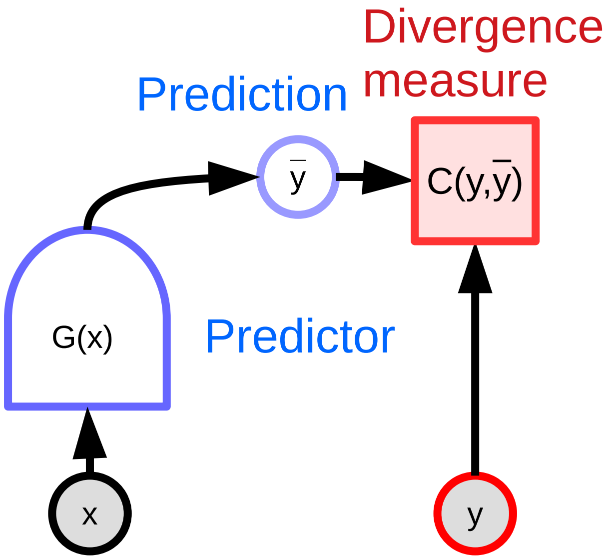
- **Example**
  - Blue dots are data points

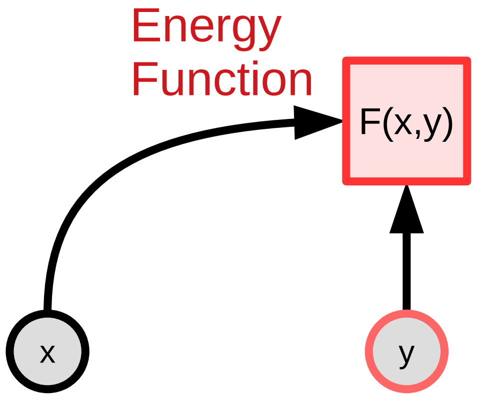


Energy-Based Model: implicit function

- **A feed-forward model is an explicit function that computes y from x.**
- **An EBM is an implicit function that captures the dependency between x and y**
- **Multiple y can be compatible with a single x**

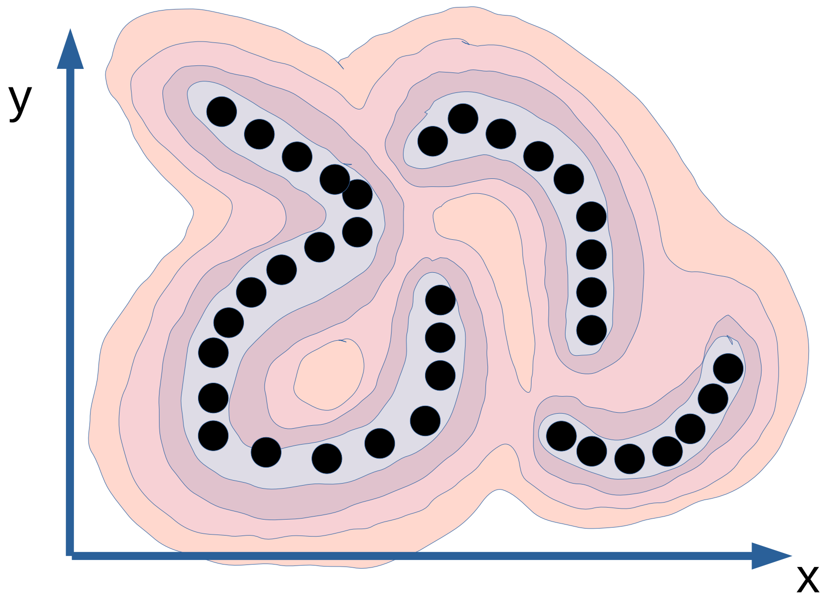




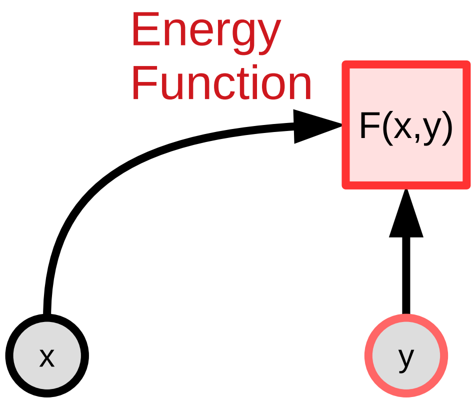


Energy-Based Model: implicit function

- **Energy function that captures the x,y dependencies:**
  - Low energy near the data points. Higher energy everywhere else.
  - If y is continuous, F should be smooth and differentiable, so we can use gradient-based inference algorithms.



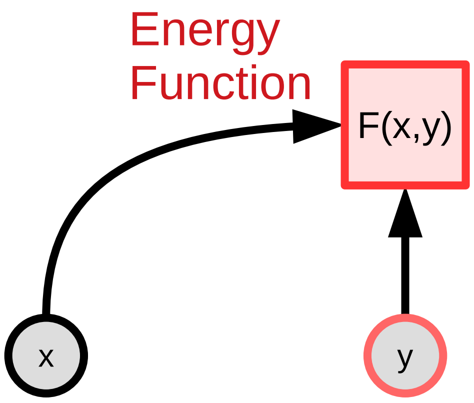
$$\check{y} = \mathrm{argmin}_y F(x,y)$$

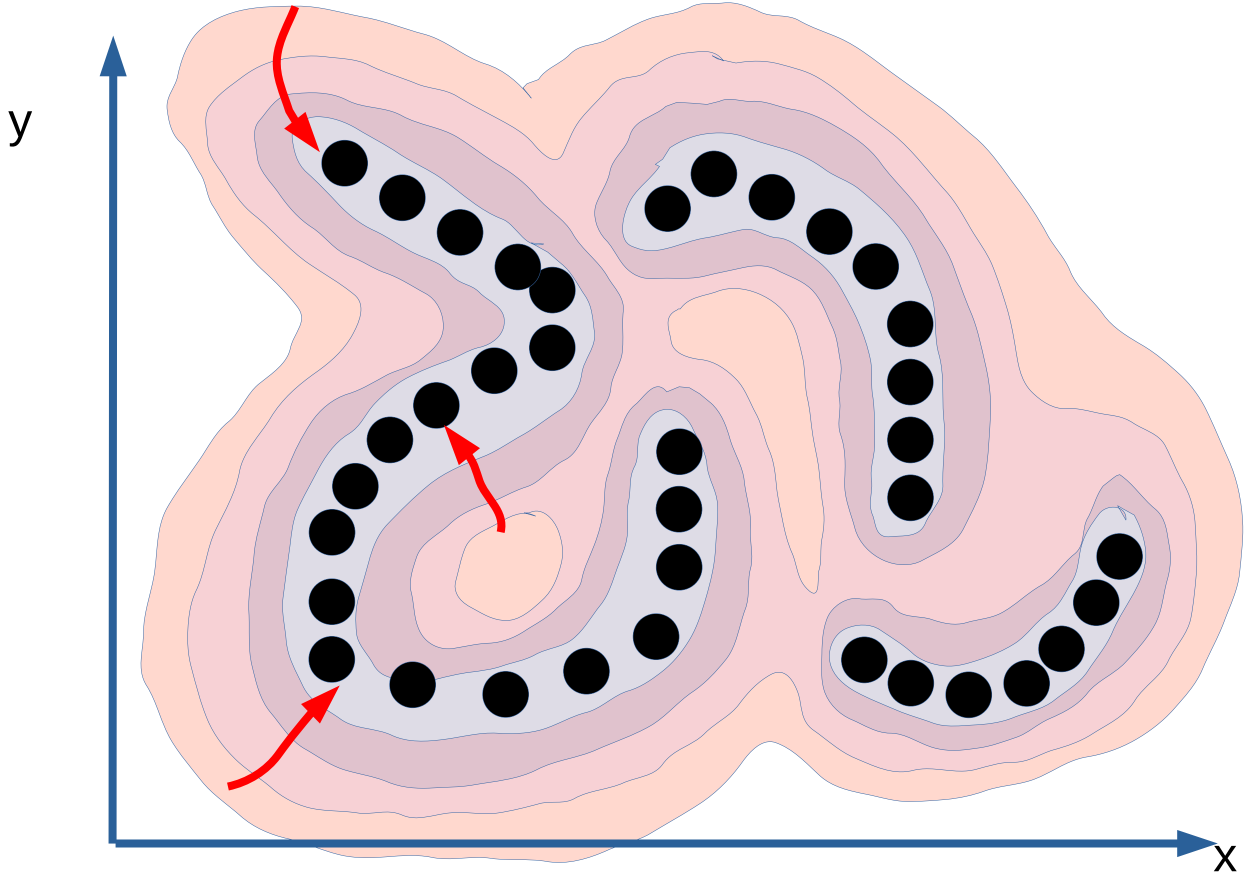


Energy-Based Model: gradient-based inference

- **If y is continuous**
  - We can use a gradient-based method for inference.

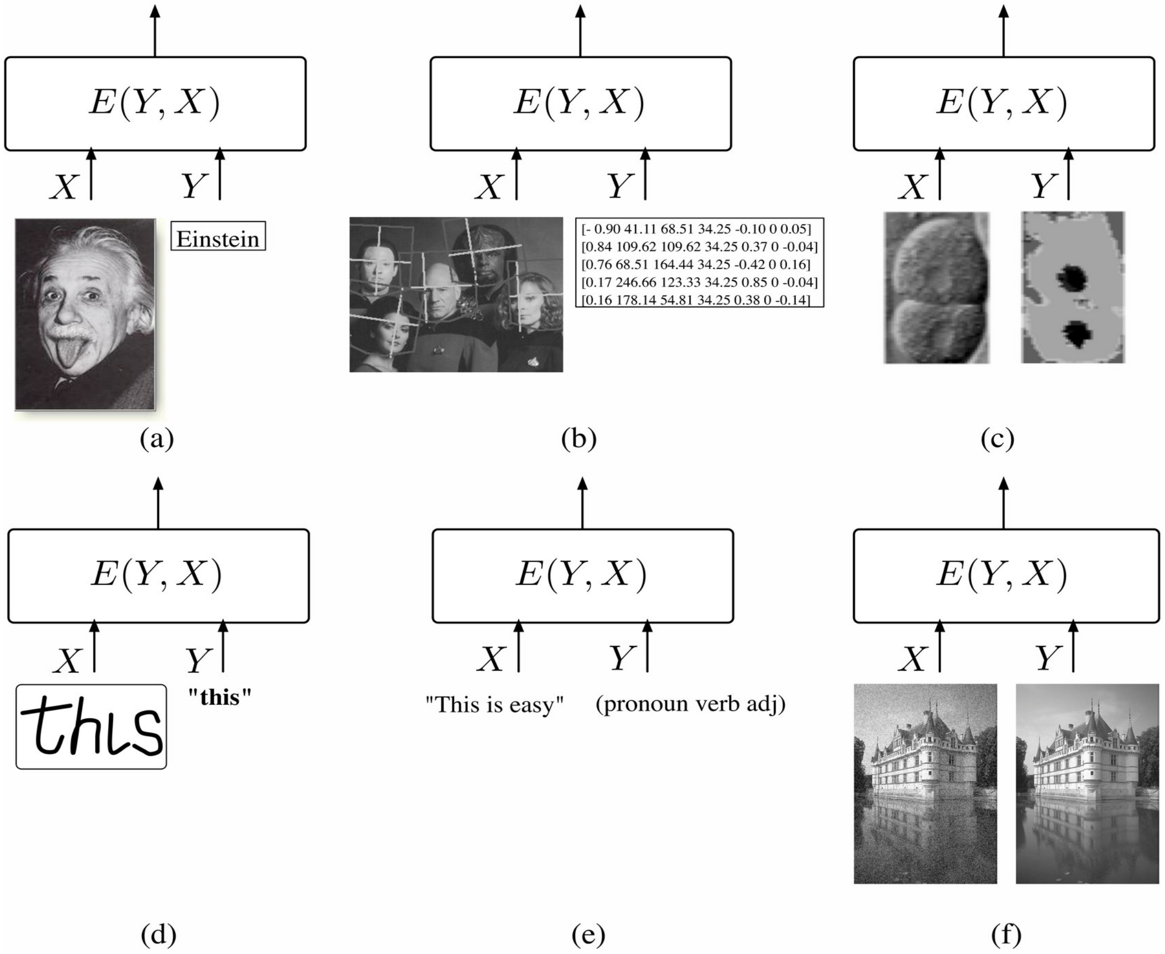
$$\check{y} = \operatorname{argmin}_y F(x, y)$$





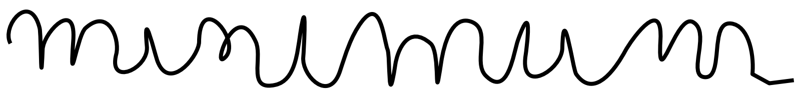
When inference is hard

- **Cases where inference is hard:**
  - Output is a high-dimensional object with structure:
    - Sequence, image, video,…
  - Output has compositional structure:
    - Text, action sequence,…
  - Output results from a long chain of reasoning
    - That can be reduced to an optimization problem

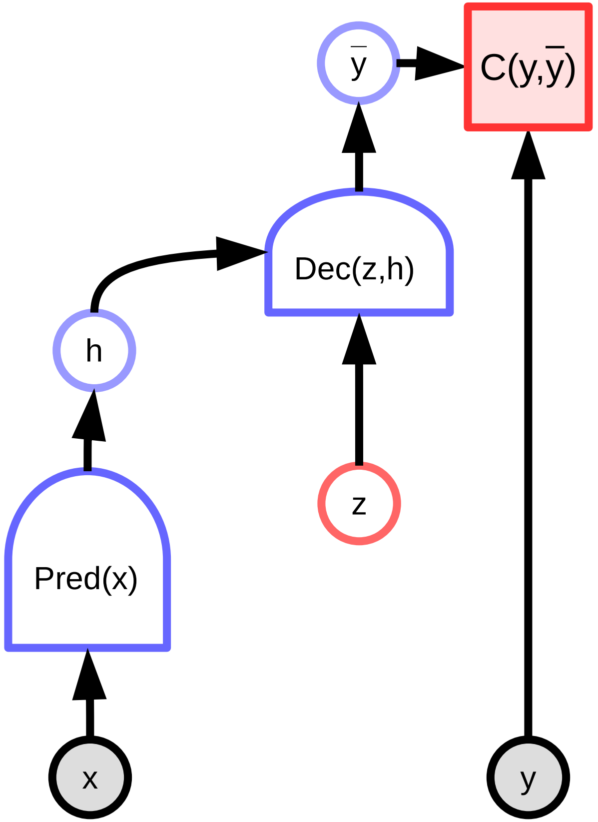


When inference involves latent variables

- **Latent variables are variables whose value is never given to us.**
  - Examples: to read a handwritten word, it helps to know where the characters are

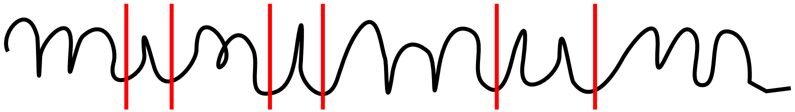


  - To recognize speech, it helps to know where the words and phonemes are
    - Youcanreadthisifyouunderstandenglish
    - Vousnepouvezpaslirececisivousneparlezpasfrançais

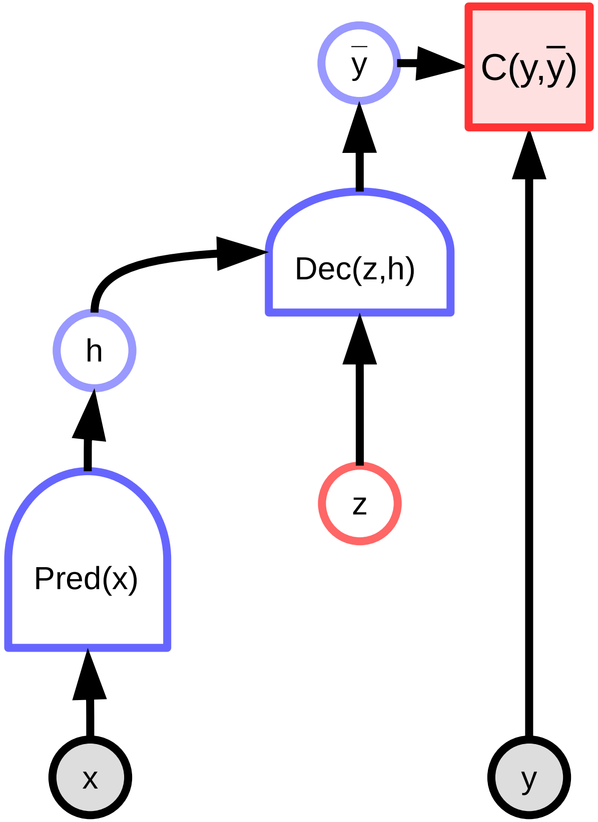


When inference involves latent variables

- **Latent variables are variables whose value is never given to us.**
  - Examples: to read a handwritten word, it helps to know where the characters are



  - To recognize speech, it helps to know where the words and phonemes are
    - You can read this if you understand english
    - Vous ne pouvez pas lire ceci si vous ne parlez pas français



Latent-Variable EBM: inference

- **Simultaneous minimization with respect to y and z**

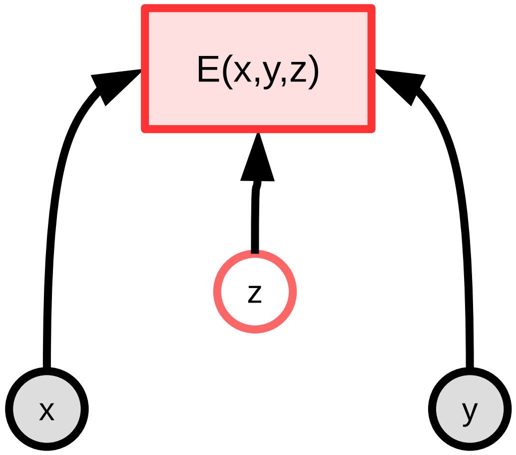
$$\check{y}, \check{z} = \text{argmin}_{y,z} E(x, y, z)$$

- **Redefinition of F(x,y)**

$$F_\infty(x, y) = \text{argmin}_z E(x, y, z)$$

$$F_\beta(x, y) = -\frac{1}{\beta} \log \int_z e^{-\beta E(x,y,z)}$$

$$\check{y} = \text{argmin}_y F(x, y)$$

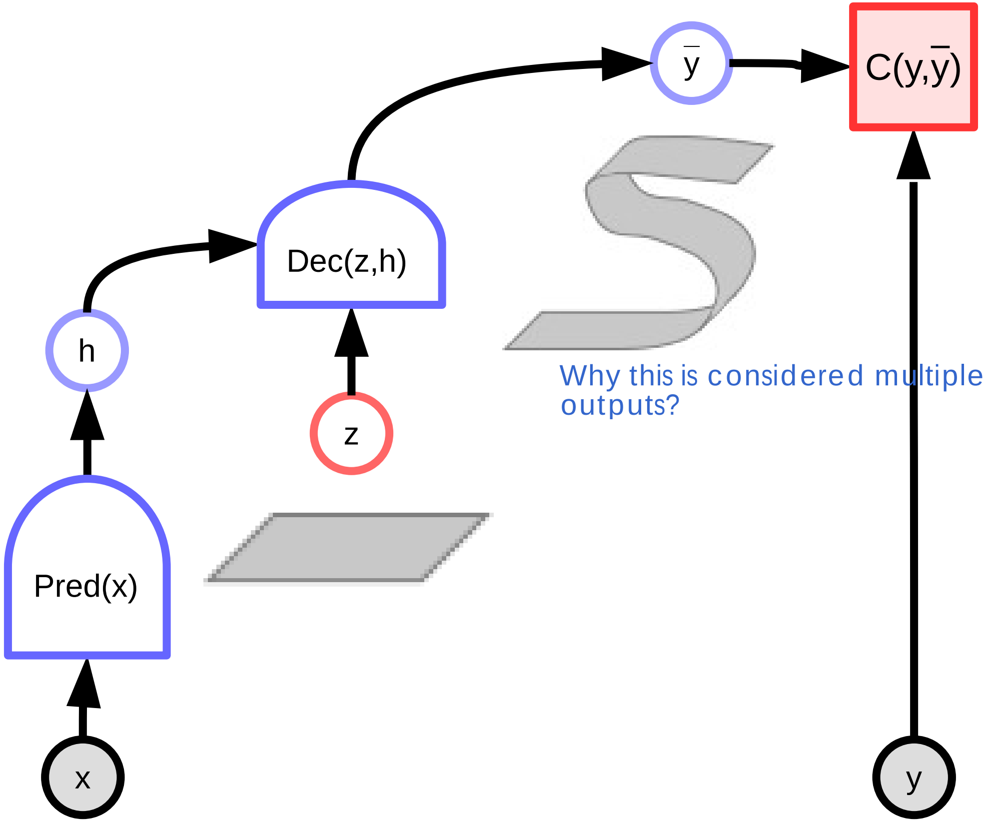


Latent-Variable EBM

- **Allowing multiple predictions through a latent variable**
- **As z varies over a set, y varies over the manifold of possible predictions**

$$F(x,y)=min_z E(x,y,z)$$

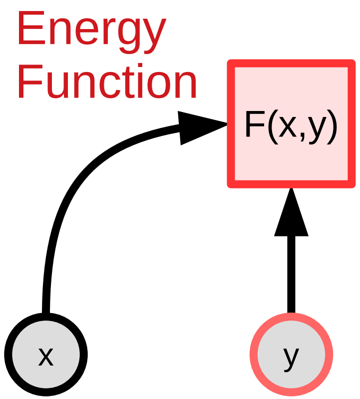
- **Useful then there are multiple correct (or plausible) outputs.**
  - Example: video prediction, text generation, translation, image synthesis….



Energy-Based Models vs Probabilistic Models

- **Probabilistic models are a special case of EBM**
  - Energies are like unnormalized negative log proabilities
- **Why use EBM instead of probabilistic models?**
  - EBM gives more flexibility in the choice of the scirring function.
  - More flexibility in the choice of objective function for learning
- **From energy to probability: Gibbs-Boltzmann distribution**
  - Beta is a positive constant

$$P(y|x) = -\frac{1}{\beta} \frac{e^{-\beta F(x,y)}}{\int_{y'} e^{-\beta F(x,y')}}$$



The larger the beta, the distribution will be more close to binary (discrete). beta is akin to inverse of temperature in Physics.

Marginalizing over the latent variable

$$P(y, z|x) = \frac{e^{-\beta E(x,y,z)}}{\int_y \int_z e^{-\beta E(x,y,z)}} \qquad P(y|x) = \int_z P(y, z|x)$$

$$P(y|x) = \frac{\int_z e^{-\beta E(x,y,z)}}{\int_y \int_z e^{-\beta E(x,y,z)}} = \frac{e^{-\beta\left[-\frac{1}{\beta}\log\int_z e^{-\beta E(x,y,z)}\right]}}{\int_y e^{-\beta\left[-\frac{1}{\beta}\log\int_z e^{-\beta E(x,y,z)}\right]}} = \frac{e^{-\beta F_\beta(x,y)}}{\int_y e^{\beta F_\beta(x,y)}}$$

- **Free energy F(x,y)** $\quad F_\beta(x, y) = -\frac{1}{\beta}\log\int_z e^{-\beta E(x,y,z)}$

FACEBOOK AI

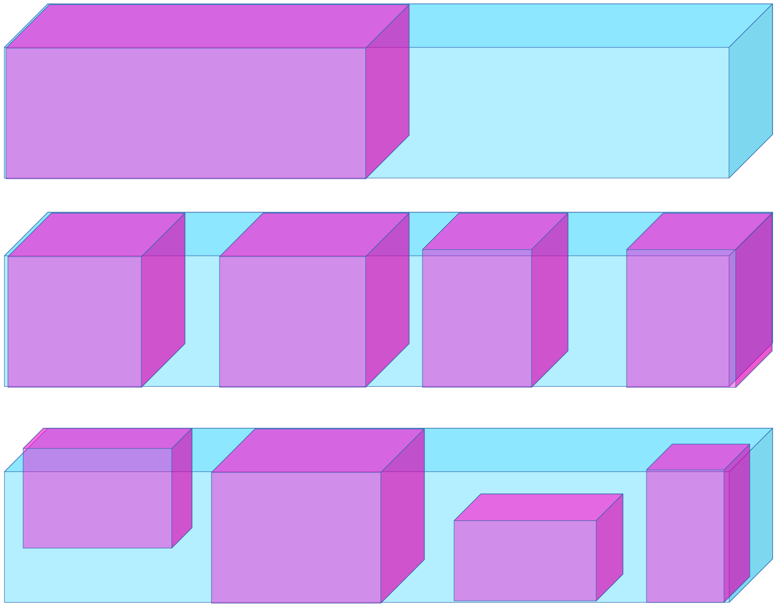
Self-Supervised Learning

Predict everything
from everything else

Self-Supervised Learning = Filling in the Blanks

- Predict any part of the input from any other part.
- Predict the future from the past.
- Predict the masked from the visible.
- Predict the any occluded part from all available parts.

time or space →



- Pretend there is a part of the input you don't know and predict that.
- Reconstruction = SSL when any part could be known or unknown

Self-Supervised Learning: filling in the bl_nks

- **Natural Language Processing: works great!**

OUTPUT: This is a **piece** of text extracted **from** a large set of **news** articles

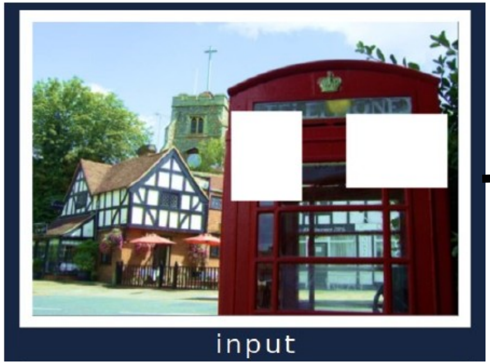
Decoder

↑ CODE

Encoder

INPUT: This is a [......] of text extracted [.....] a large set of [......] articles

- **Image Recognition / Understanding: works so-so** [Pathak et al 2014]



→ Encoder → CODE → Decoder →

Learning Representations through Pretext SSL Tasks

- **Text / symbol sequences (discrete, works great!)**
  - Future word(s) prediction (NLM)
  - Masked words prediction (BERT et al.)
- **Image (continuous)**
  - Inpainting, colorization, super-resolution
- **Video (continuous)**
  - Future frame(s) prediction
  - Masked frames prediction
- **Signal / Audio (continuous)**
  - Restoration
  - Future prediction

Self-Supervised Learning works **very** well for text

- **Word2vec**
  - [Mikolov 2013]
- **FastText**
  - [Joulin 2016] (FAIR)
- **BERT**
  - Bidirectional Encoder Representations from Transformers
  - [Devlin 2018]
- **Cloze-Driven Auto-Encoder**
  - [Baevski 2019] (FAIR)
- **RoBERTa** [Ott 2019] (FAIR)

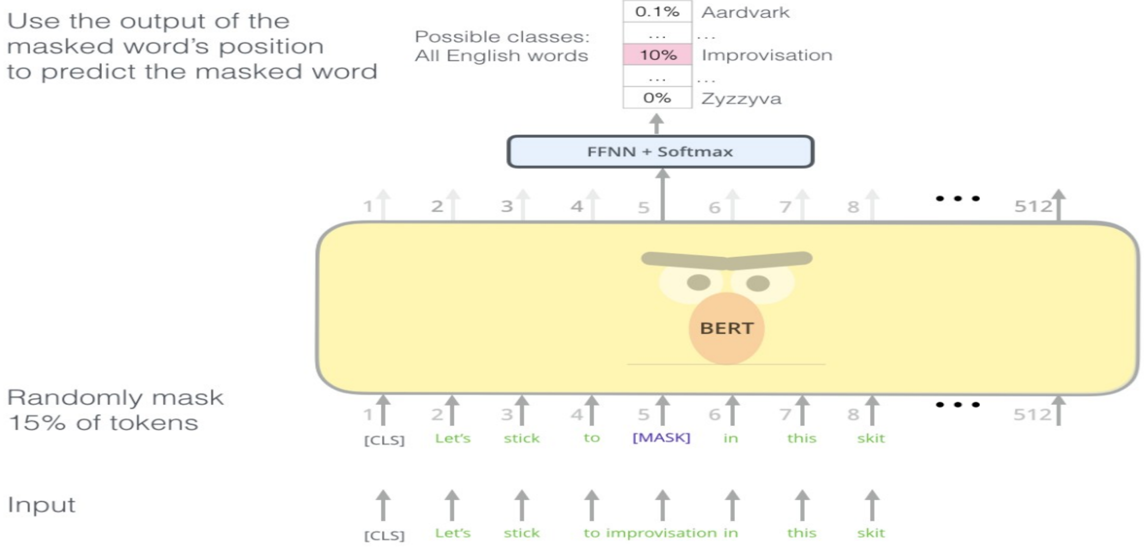
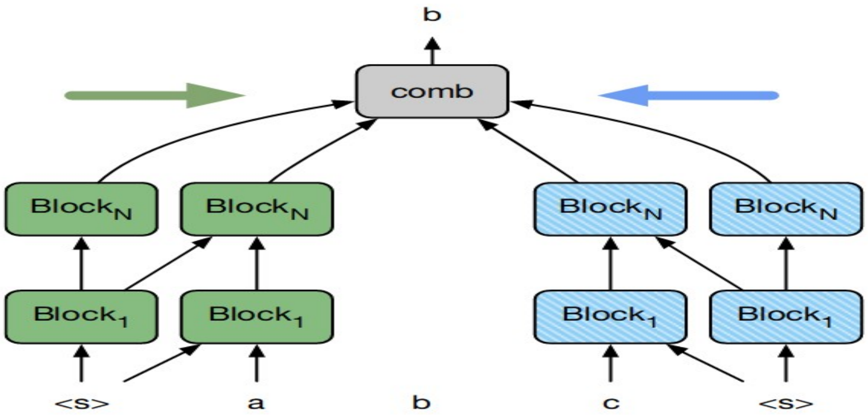


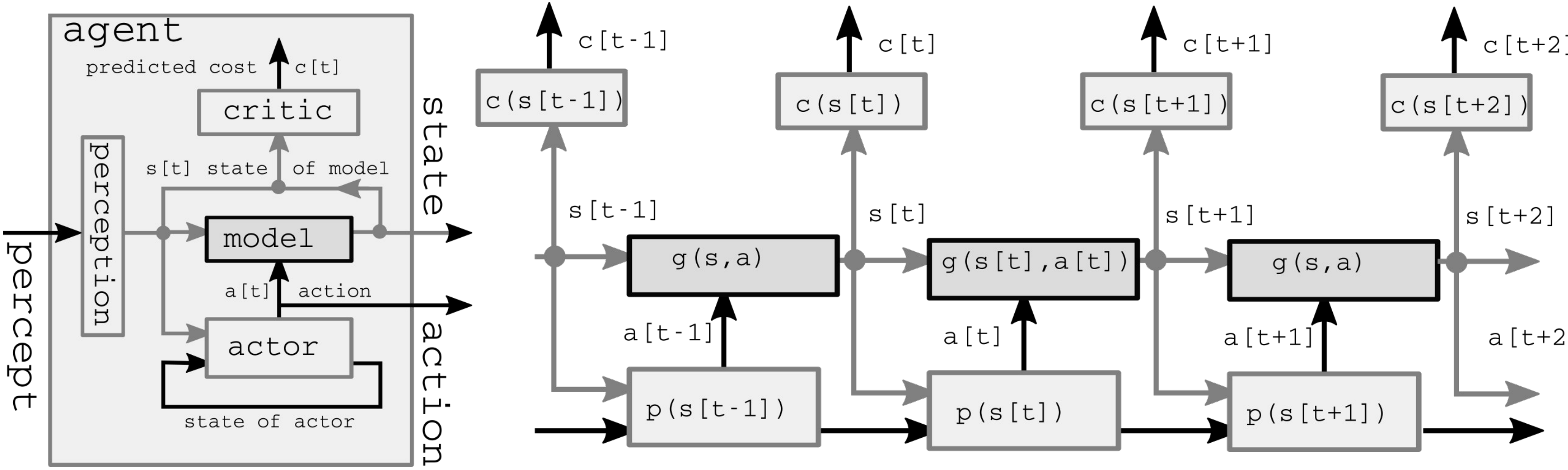
Figure credit: Jay Alammar http://jalammar.github.io/illustrated-bert/



SSL works less well for images and video

Learning World Models for Autonomous AI Agents

- **Learning forward models for control**
  - s[t+1] = g(s[t], a[t], z[t])
  - Model-predictive control, model-predictive policy learning, model-based RL
  - Robotics, games, dialog, HCI, etc

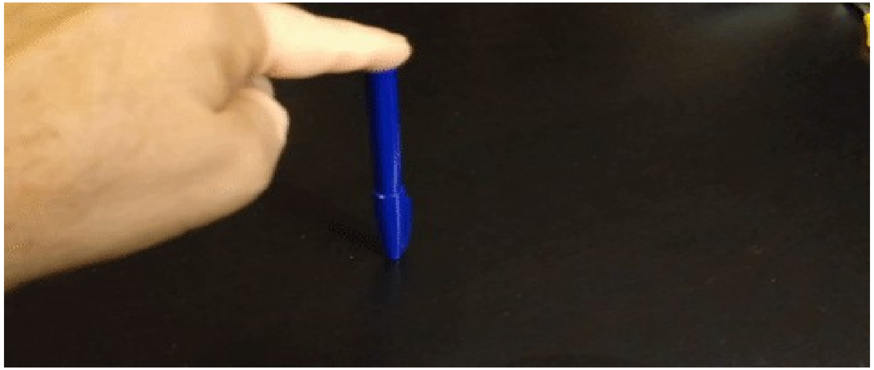


Self-Supervised Learning for Video Prediction

- **The world is not entirely predictable**
- **There are many plausible continuations to a video segment**

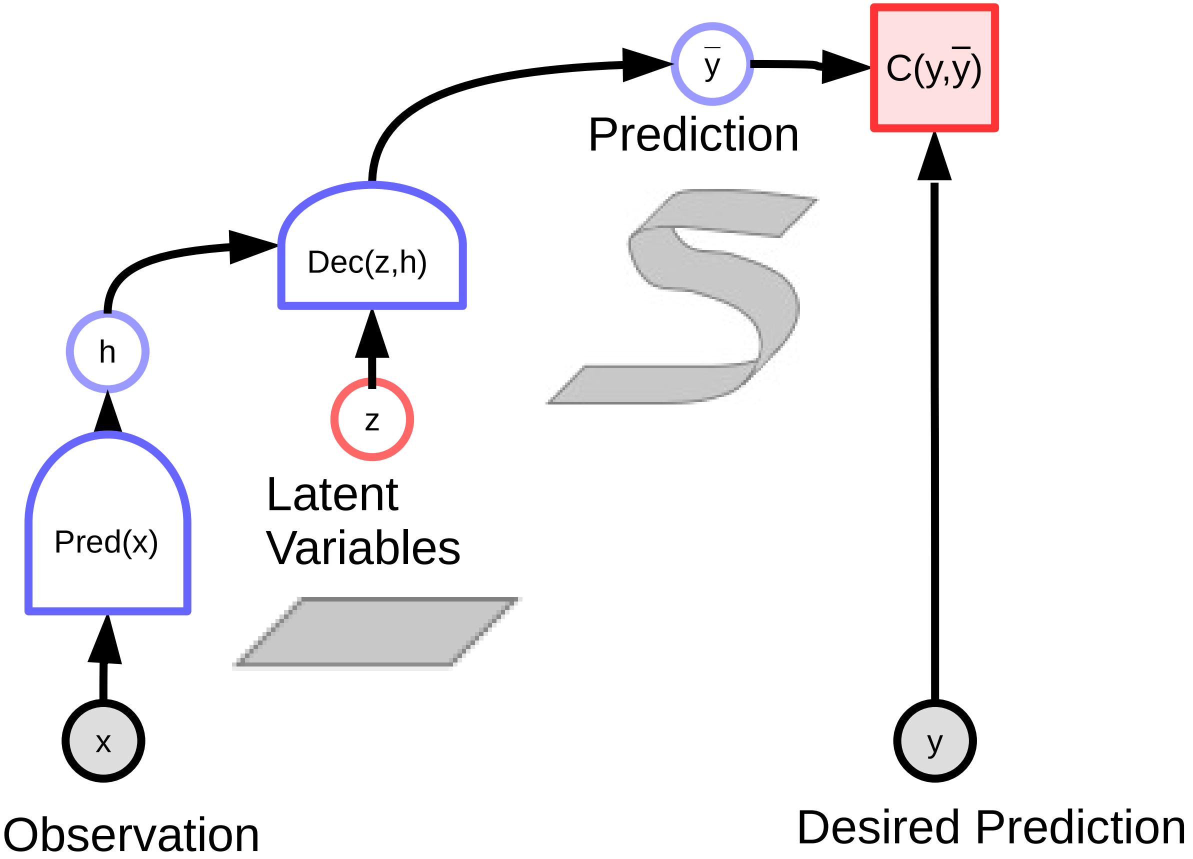
The world is stochastic

- Training a system to make a single prediction makes it predict the average of all plausible predictions
- Blurry predictions!



Solution: latent variable energy-based models

- **Latent variables allows system to make multiple predictions**



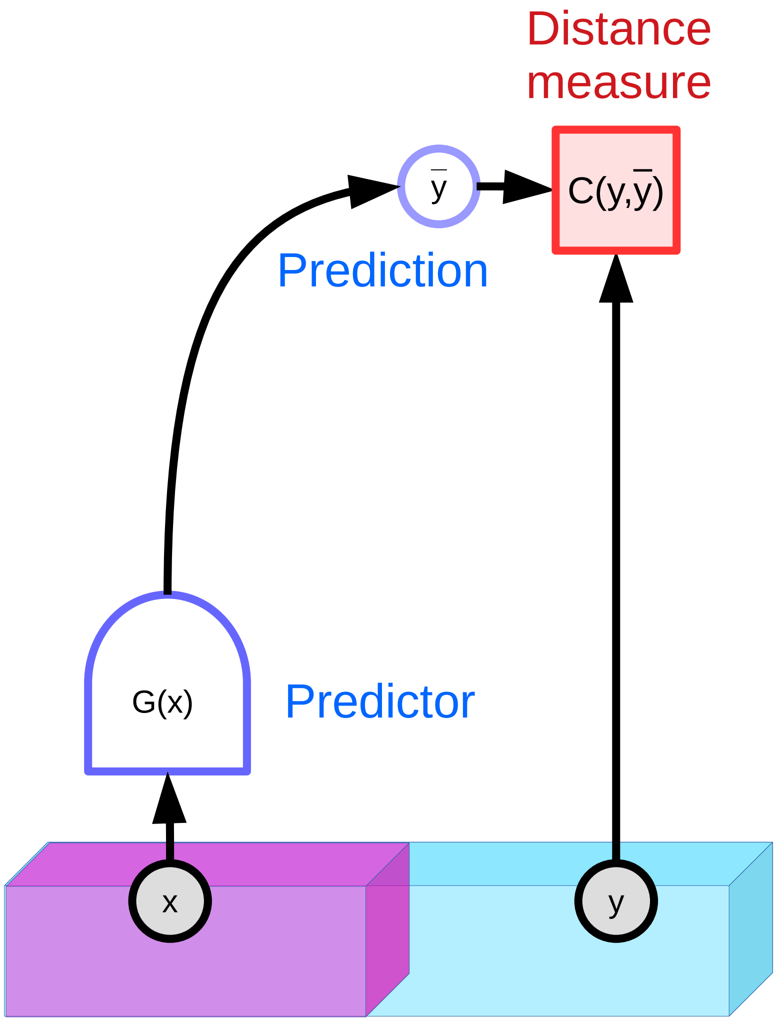
Self-supervised Adversarial Learning for Video Prediction

- **Our brains are “prediction machines”**
- **Can we train machines to predict the future?**
- **Some success with “adversarial training”**
  - [Mathieu, Couprie, LeCun arXiv:1511:05440]
- **But we are far from a complete solution.**

Problem: uncertainty!

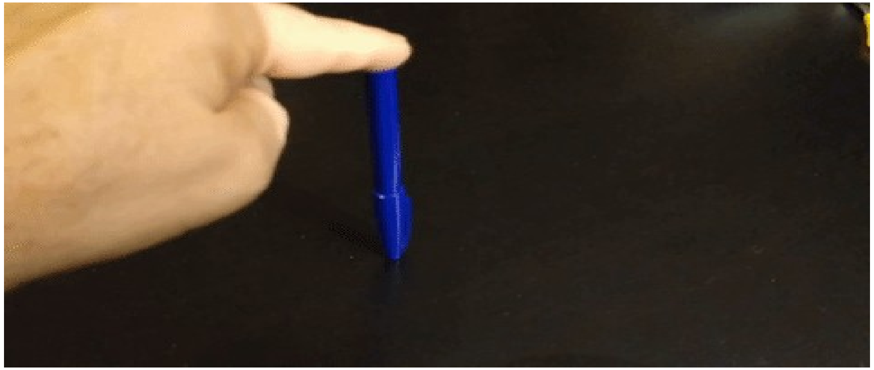
- **There are many plausible words that complete a text.**
- **There are infinitely many plausible frames to complete a video.**
- **Deterministic predictors don't work!**
- **How to deal with uncertainty in the prediction?**

$$E(x,y)=C(y,G(x))$$



The world is not entirely predictable / stochastic

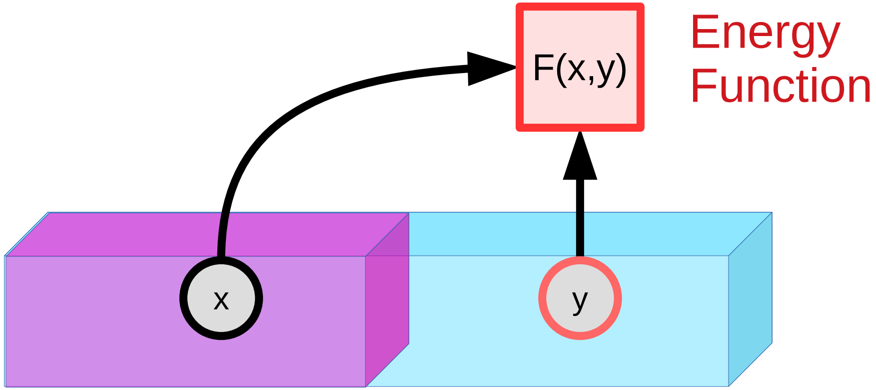
- **Video prediction:**
  - A deterministic predictor with L2 distance will predict the average of all plausible futures.
- **Blurry prediction!**



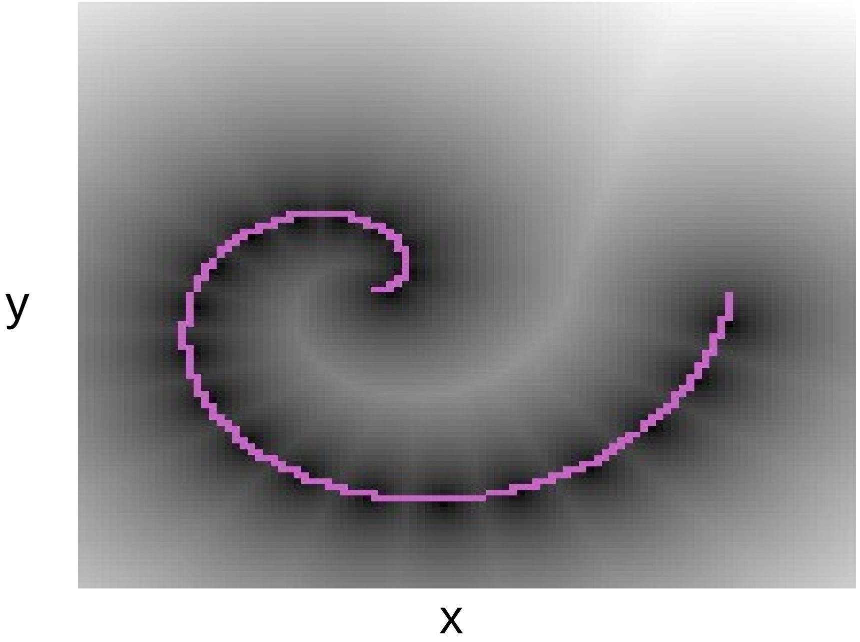
Energy-Based Model

- **Scalar-valued energy function: F(x,y)**
  - measures the compatibility between x and y
  - Low energy: y is good prediction from x
  - High energy: y is bad prediction from x

- Inference: $\check{y} = argmin_y F(x,y)$



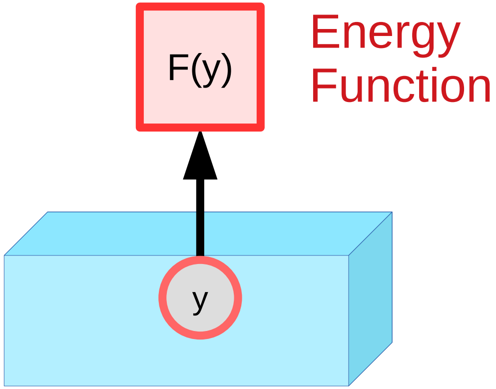
Dark = low energy (good)
Bright = high energy (bad)
Purple = data manifold



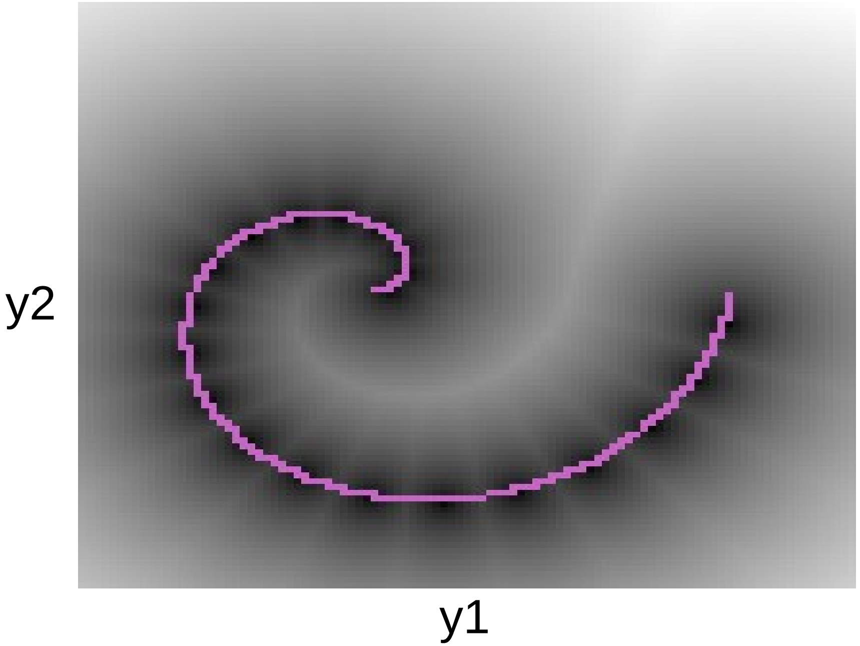
[Figure from M-A Ranzato's PhD thesis]

Energy-Based Model: unconditional version

- **Scalar-valued energy function: F(y)**
  - measures the compatibility between the components of y
  - If we don't know in advance which part of y is known and which part is unknown
  - Example: auto-encoders, generative models (energy = -log likelihood)

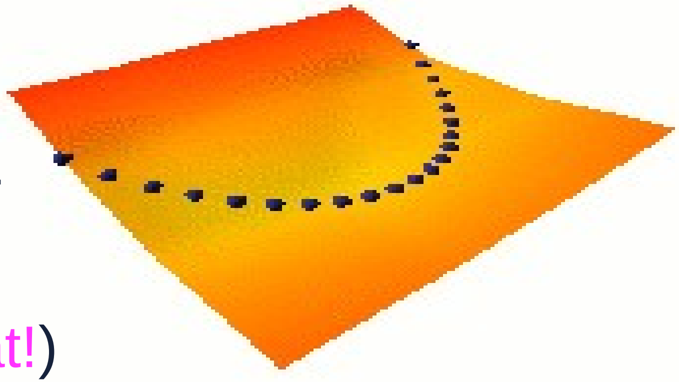


Dark = low energy (good)
Bright = high energy (bad)
Purple = data manifold

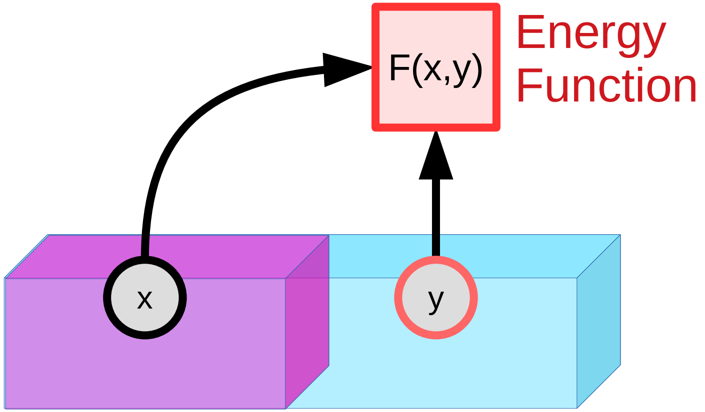


Training an Energy-Based Model

- **Parameterize F(x,y)**
- **Get training data (x[i], y[i])**
- **Shape F(x,y) so that:**
  - F(x[i], y[i]) is strictly smaller than F(x[i], y) for all y different from y[i]
  - F is smooth (probabilistic methods break that!)



- **Two classes of learning methods:**
  - 1. **Contrastive methods:** push down on F(x[i], y[i]), push up on other points F(x[i], y')
  - 2. **Architectural Methods:** build F(x,y) so that the volume of low energy regions is limited or minimized through regularization



Seven Strategies to Shape the Energy Function

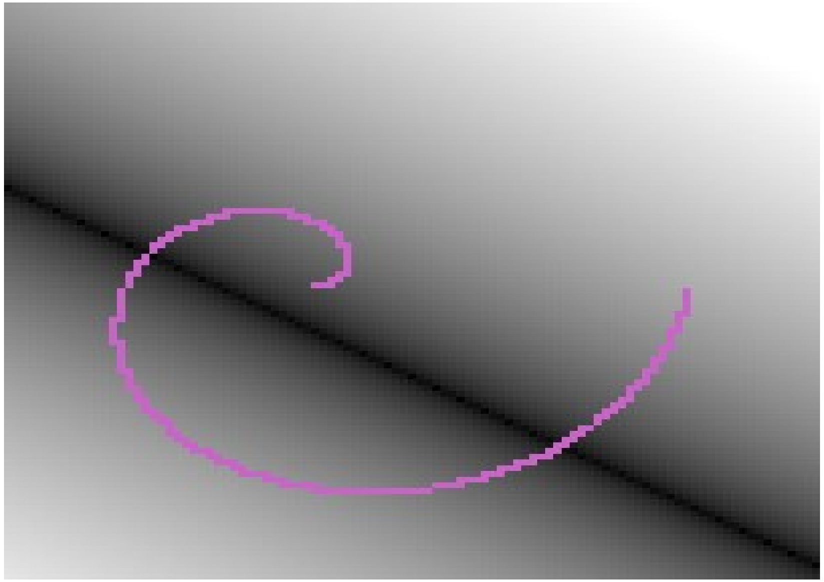
- **Contrastive: [they all are different ways to pick which points to push up]**
  - C1: push down of the energy of data points, push up everywhere else: Max likelihood (needs tractable partition function or variational approximation)
  - C2: push down of the energy of data points, push up on chosen locations: max likelihood with MC/MMC/HMC, Contrastive divergence, Metric learning, Ratio Matching, Noise Contrastive Estimation, Min Probability Flow, adversarial generator/GANs
  - C3: train a function that maps points off the data manifold to points on the data manifold: denoising auto-encoder, masked auto-encoder (e.g. BERT)
- **Architectural: [they all are different ways to limit the information capacity of the code]**
  - A1: build the machine so that the volume of low energy stuff is bounded: PCA, K-means, Gaussian Mixture Model, Square ICA…
  - A2: use a regularization term that measures the volume of space that has low energy: Sparse coding, sparse auto-encoder, LISTA, Variational auto-encoders
  - A3: F(x,y) = C(y, G(x,y)), make G(x,y) as "constant" as possible with respect to y: Contracting auto-encoder, saturating auto-encoder
  - A4: minimize the gradient and maximize the curvature around data points: score matching

Simple examples: PCA and K-means

- **Limit the capacity of z so that the volume of low energy stuff is bounded**
  - PCA, K-means, GMM, square ICA...

PCA: z is low dimensional

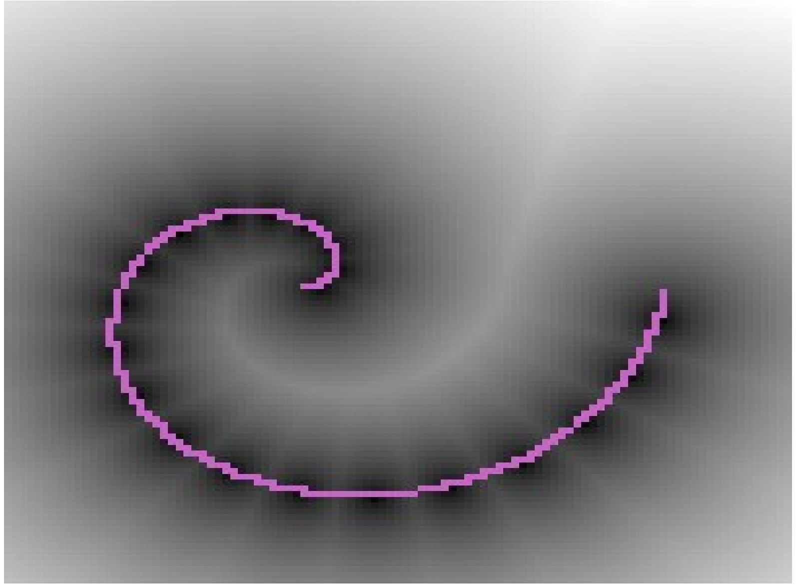
$$F(Y) = ||W^T W Y - Y||^2$$



K-Means,

Z constrained to 1-of-K code

$$F(Y) = min_z \sum_i ||Y - W_i Z_i||^2$$

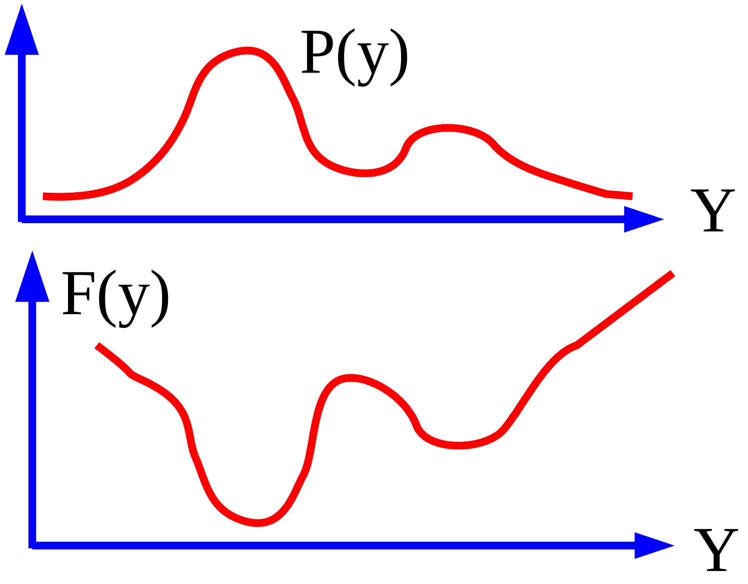


Familiar Example: Maximum Likelihood Learning

- **The energy can be interpreted as an unnormalized negative log density**
- **Gibbs distribution: Probability proportional to exp(-energy)**
  - Beta parameter is akin to an inverse temperature
- **Don't compute probabilities unless you absolutely have to**
  - Because the denominator is often intractable

$$P(y) = \frac{\exp[-\beta F(y)]}{\int_{y'} \exp[-\beta F(y')]}$$

$$P(y|x) = \frac{\exp[-\beta F(x,y)]}{\int_{y'} \exp[-\beta F(x,y')]}$$



push down of the energy of data points, push up everywhere else

- Max likelihood (requires a tractable partition function)

Maximizing P(Y|W) on training samples

make this big

$$P(Y|W) = \frac{e^{-\beta E(Y,W)}}{\int_y e^{-\beta E(y,W)}}$$

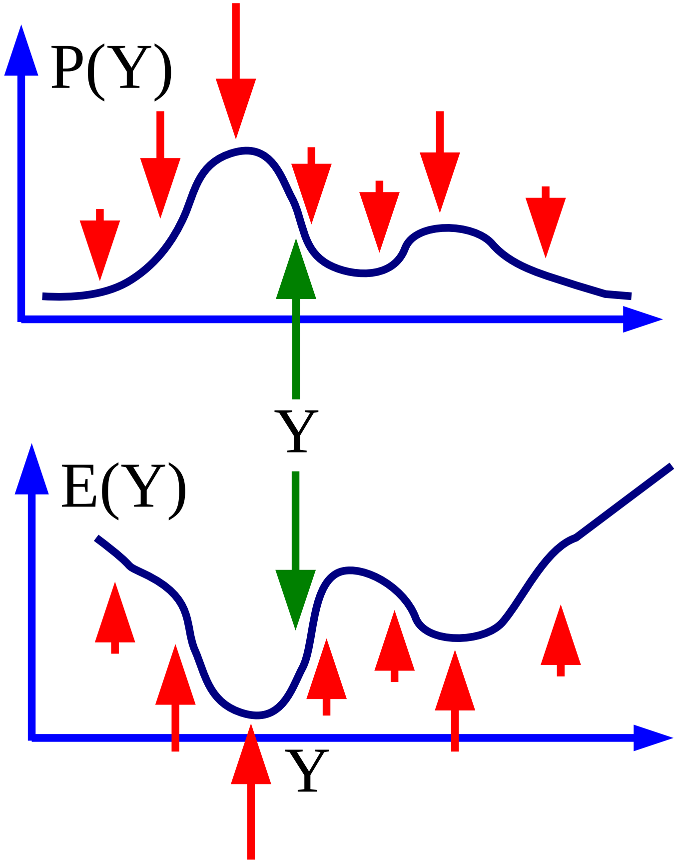
make this small

Minimizing -log P(Y,W) on training samples

$$L(Y,W) = E(Y,W) + \frac{1}{\beta} \log \int_y e^{-\beta E(y,W)}$$

make this small

make this big



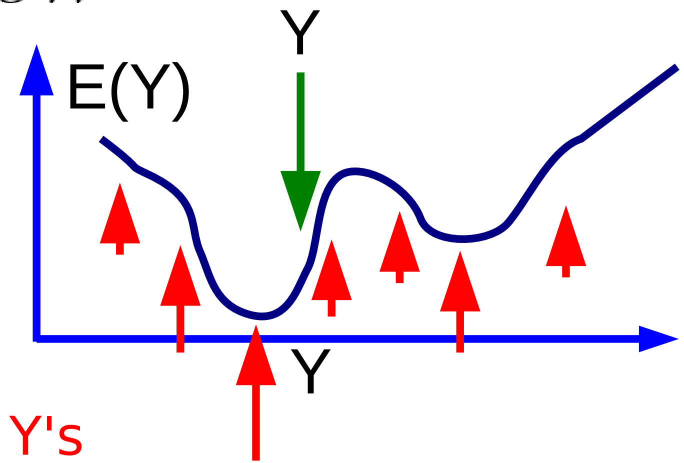
push down of the energy of data points, push up everywhere else

Gradient of the negative log-likelihood loss for one sample Y:

$$\frac{\partial L(Y,W)}{\partial W} = \frac{\partial E(Y,W)}{\partial W} - \int_y P(y|W) \frac{\partial E(y,W)}{\partial W}$$

Gradient descent:

$$W \leftarrow W - \eta \frac{\partial L(Y,W)}{\partial W}$$



Pushes down on the energy of the samples

Pulls up on the energy of low-energy Y's

$$W \leftarrow W - \eta \frac{\partial E(Y,W)}{\partial W} + \eta \int_y P(y|W) \frac{\partial E(y,W)}{\partial W}$$

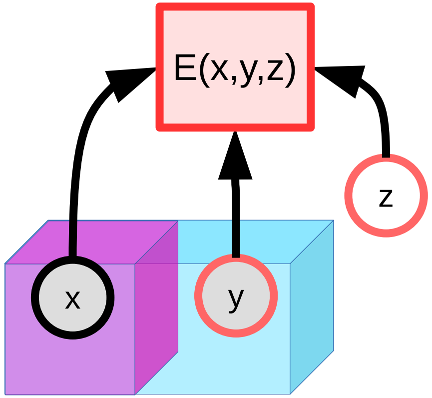
Latent-Variable EBM

- **Allowing multiple predictions through a latent variable**

- **Conditional:**

$$F(x,y)=min_z E(x,y,z)$$

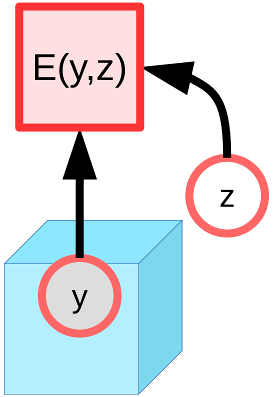
$$F(x,y)=-\frac{1}{\beta}\log\left[\int_z \exp(-\beta E(x,y,z))\right]$$



- **Unconditional**

$$F(y)=min_z E(y,z)$$

$$F(y)=-\frac{1}{\beta}\log\left[\int_z \exp(-\beta E(y,z))\right]$$



Latent-Variable EBM for multimodal prediction

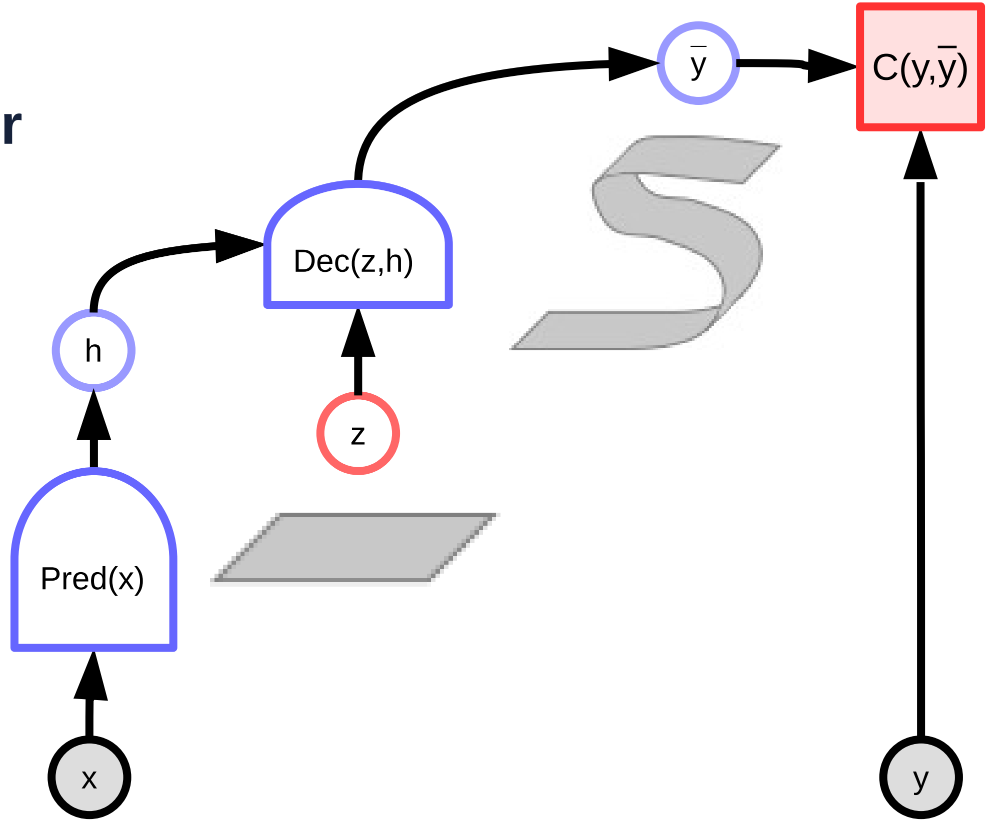
- **Allowing multiple predictions through a latent variable**

- **As z varies over a set, y varies over the manifold of possible predictions**

$$F(x,y) = min_z E(x,y,z)$$

- **Examples:**
  - K-means
  - Sparse modeling
  - GLO

    [Bojanowski arXiv:1707.05776]

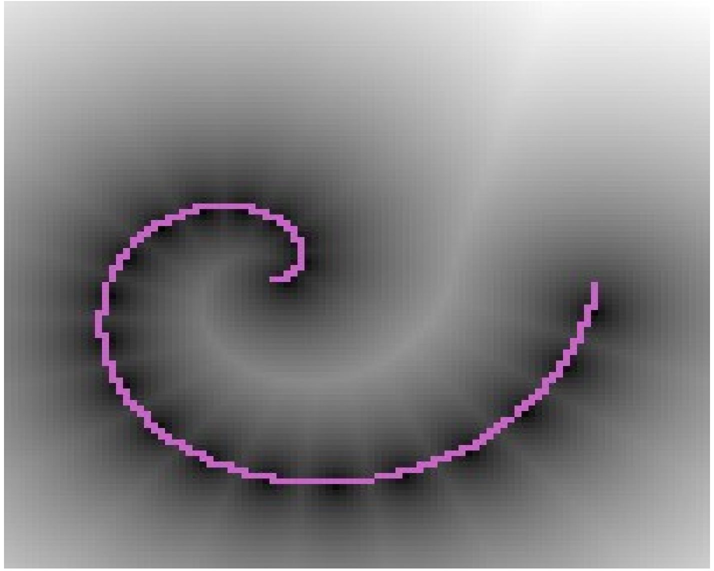


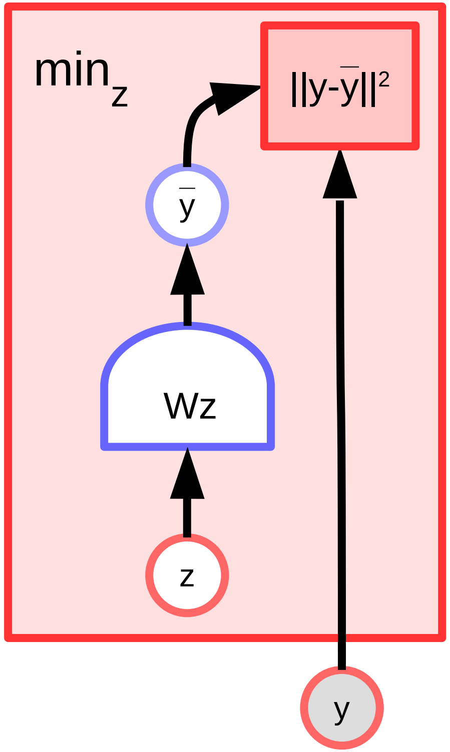
Latent-Variable EBM example: K-means

- **Decoder is linear, z is a 1-hot vector (discrete)**
- **Energy function:** $E(y,z) = ||y - Wz||^2 \quad z \in 1\,hot$
- **Inference by exhaustive search**

$$F(y) = min_z E(y,z)$$

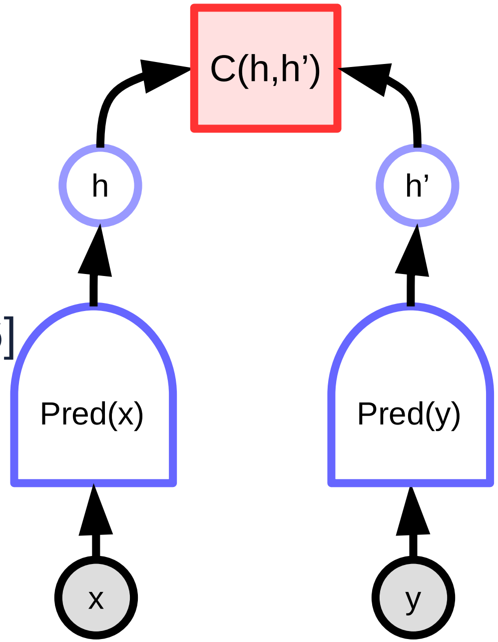
- **Volume of low-energy regions limited by number of prototypes k**





Contrastive Embedding

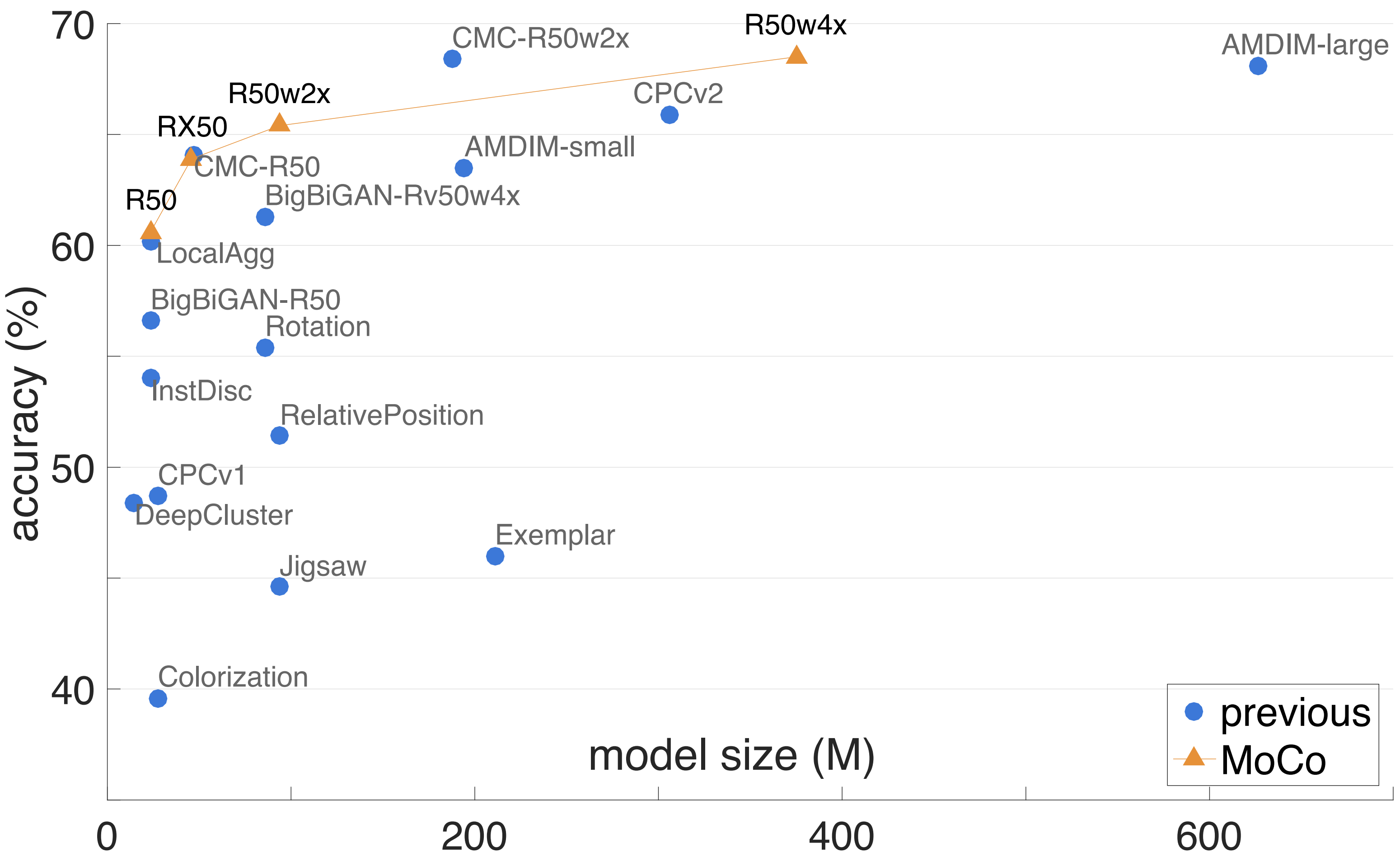
- **Distance measured in feature space**
- **Multiple “predictions” through feature invariance**
- **Siamese nets, metric learning** [YLC NIPS’93,CVPR’05,CVPR’06]
- **Advantage: no pixel-level reconstruction**
- **Difficulty: hard negative mining**
- **Successful examples for images:**
  - DeepFace [Taigman et al. CVPR’14]
  - PIRL [Misra et al. To appear]
  - MoCo [He et al. Arxiv:1911.05722]
- **Video / Audio**
  - Temporal proximity [Taylor CVPR’11]
  - Slow feature [Goroshin NIPS’15]



Positive pair:
Make F small

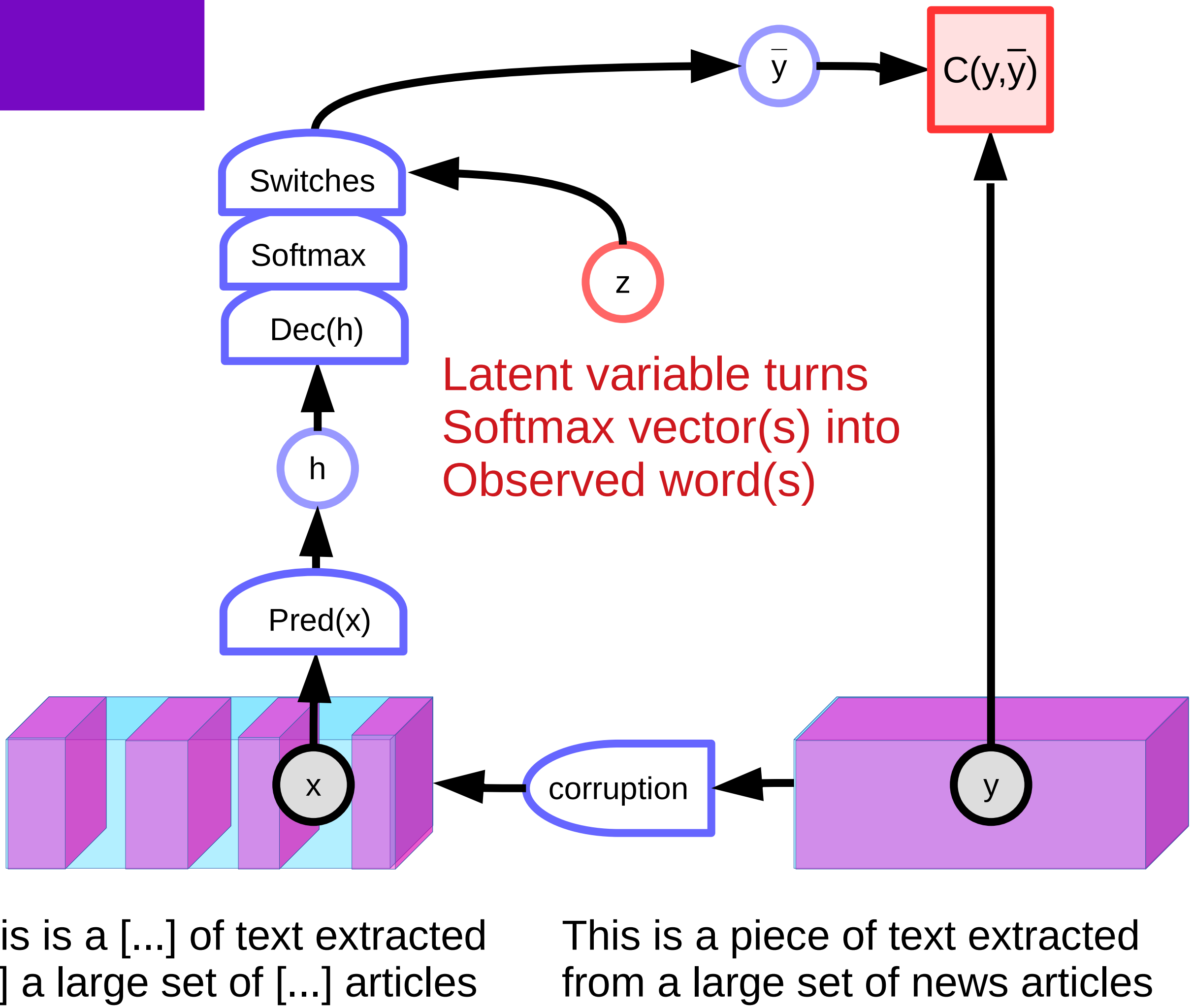
Negative pair:
Make F large

MoCo on ImageNet [He et al. Arxiv:1911.05722]



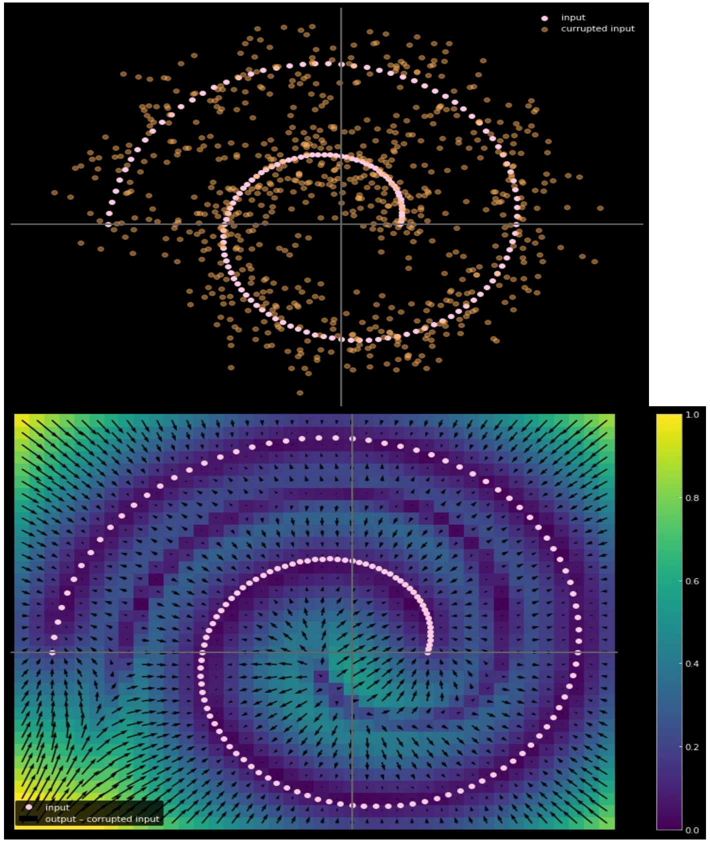
Denoising AE: discrete

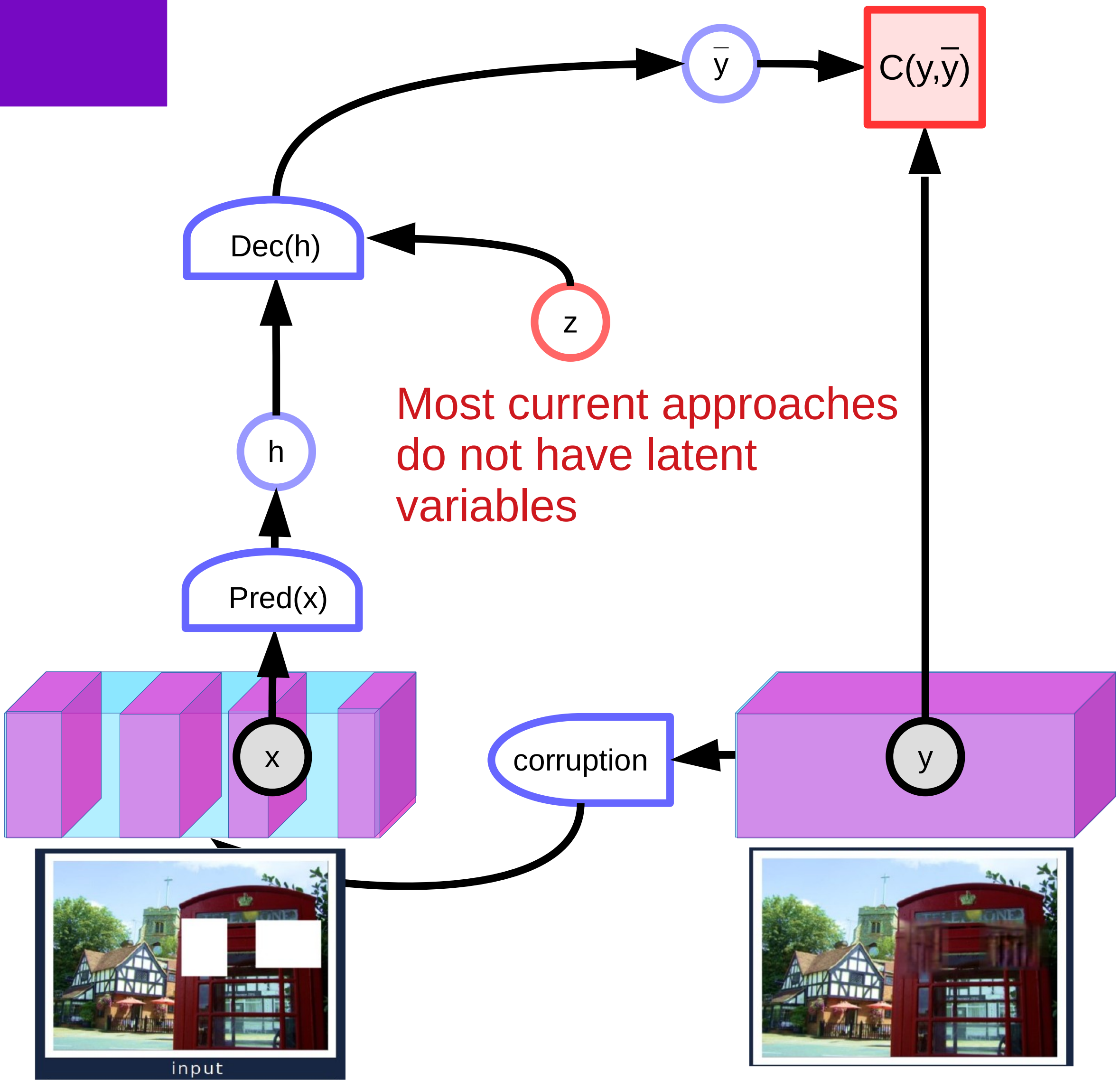
- **[Vincent et al. JMLR 2008]**
- **Masked Auto-Encoder**
  - [BERT et al.]
- **Issues:**
  - latent variables are in output space
  - No abstract LV to control the output
  - How to cover the space of corruptions?



Denoising AE: continuous

- **Image inpainting** [Pathak 17]
- Latent variables? GAN?



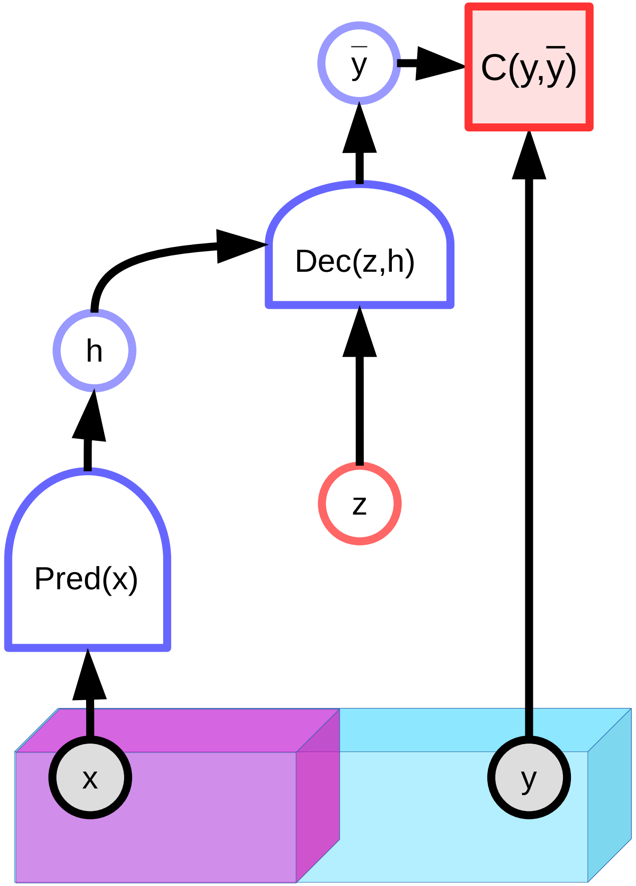


Prediction with Latent Variables

- **If the Latent has too much capacity...**
  - e.g. if it has the same dimension as y
- **... then the entire y space could be perfectly reconstructed**

$$E(x,y,z)=C(y,Dec(Pred(x),z))$$

- **For every y, there is always a z that will reconstruct it perfectly**
  - The energy function would be zero everywhere
  - This is no a good model….
- **Solution: limiting the information capacity of the latent variable z.**

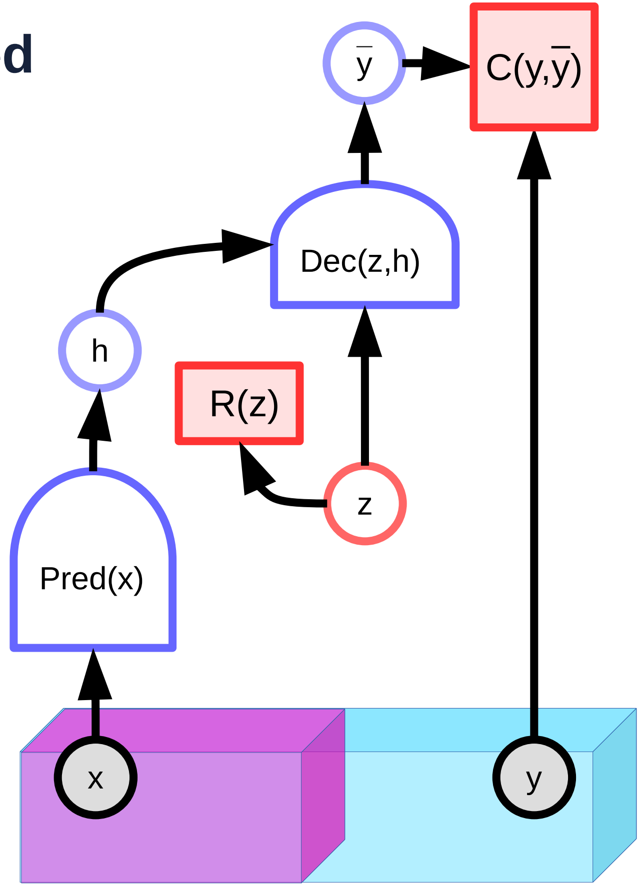


Regularized Latent Variable EBM

- **Regularizer R(z) limits the information capacity of z**
- **Without regularization, every y may be reconstructed exactly (flat energy surface)**

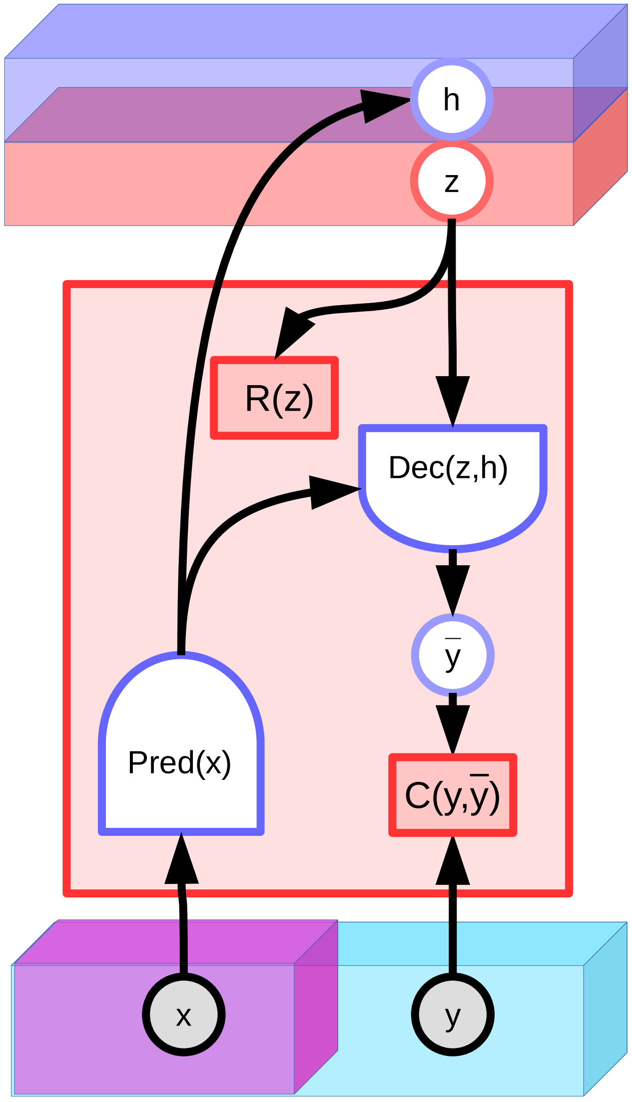
$$E(x,y,z)=C(y,Dec(Pred(x),z))+\lambda R(z)$$

- **Examples of R(z):**
  - Effective dimension
  - Quantization / discretization
  - L0 norm (# of non-0 components)
  - L1 norm with decoder normalization
  - Maximize lateral inhibition / competition
  - Add noise to z while limiting its L2 norm (VAE)
  - <your_information_throttling_method_goes_here>



Sequence → Abstract Features

- **Regularized LV EBM is passed over a sequence (e.g. a video, audio, text)**
- **The sequence of corresponding h and z is collected**
  - It contains all the information about the input sequence
  - h contains the information in x that is useful to predict y
  - z contains the complementary information, not present in x or h.
- **Several such SSL modules can be stacked to learn hierarchical representations of sequences**

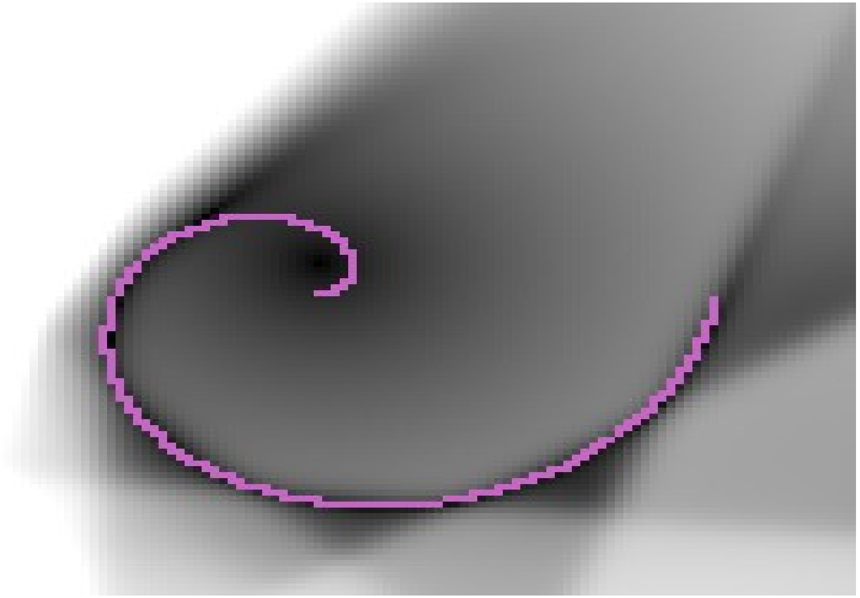


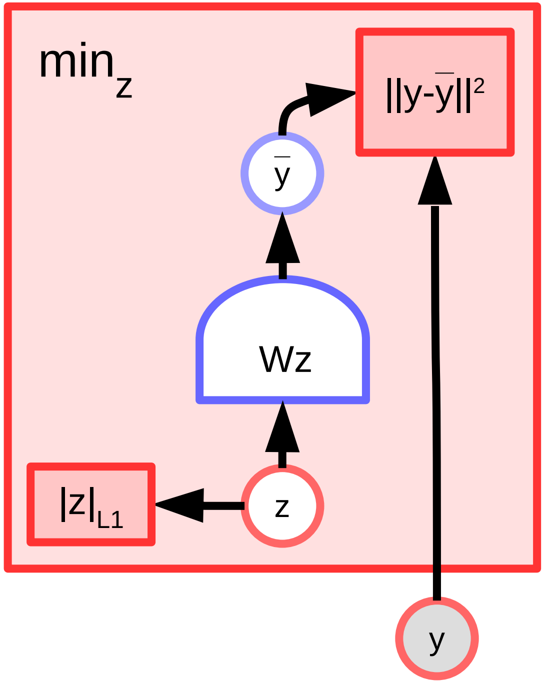
Y. LeCun

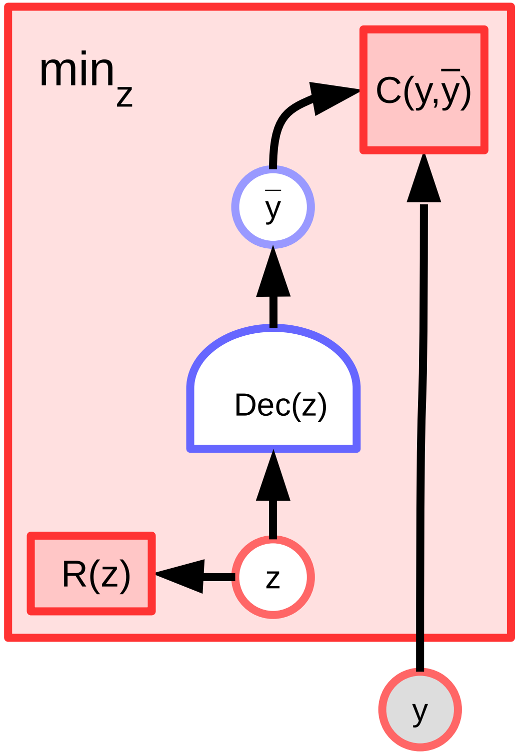
Unconditional Regularized Latent Variable EBM

- **Unconditional form. Reconstruction. No x, no predictor.**
- **Example: sparse modeling**
  - Linear decoder
  - L1 regularizer on Z

$$E(y,z) = \|y - Wz\|^2 + \lambda |z|$$





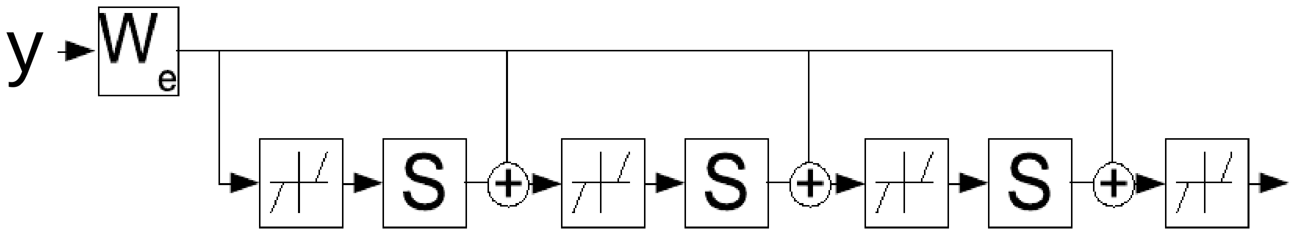


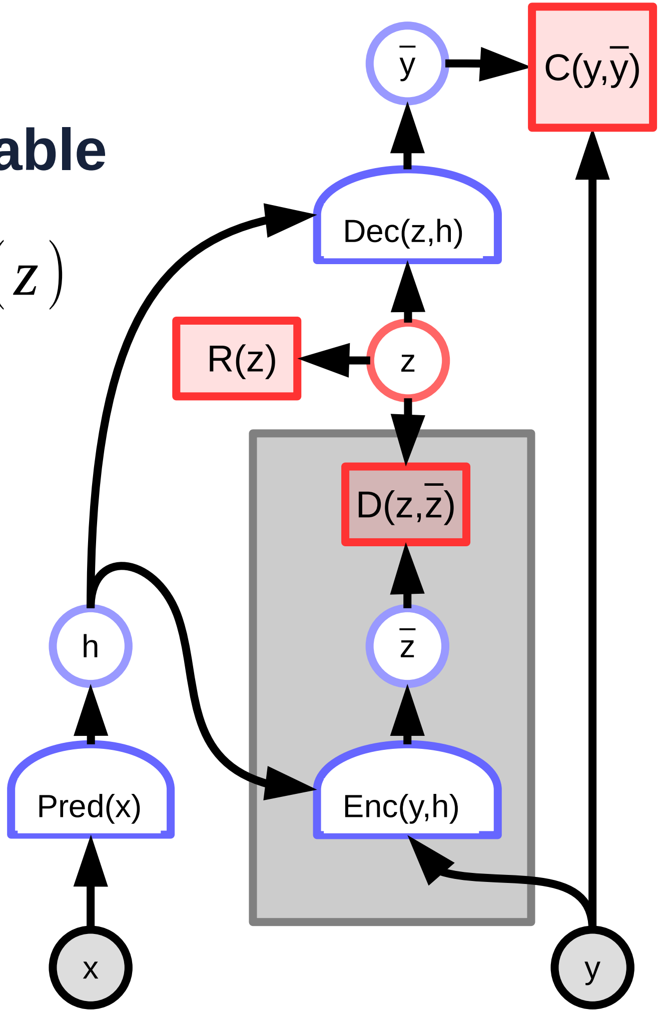
LatVar inference is expensive!

- **Let's train an encoder to predict the latent variable**

$$E(x,y,z)=C(y,Dec(z,h))+D(z,Enc(x,y))+\lambda R(z)$$

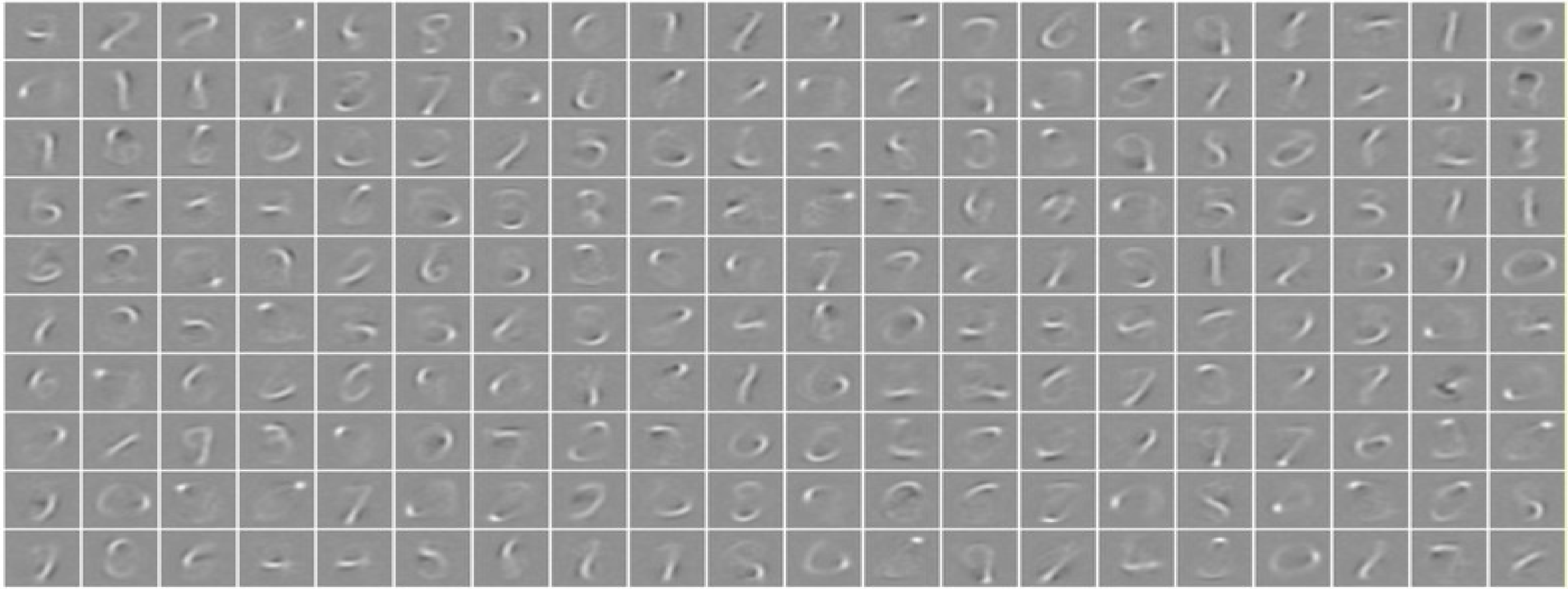
- **Predictive Sparse Modeling**
  - R(z) = L1 norm of z
  - Dec(z,h) gain must be bounded (clipped weights)
  - Sparse Auto-Encoder
  - LISTA [Gregor ICML 2010]





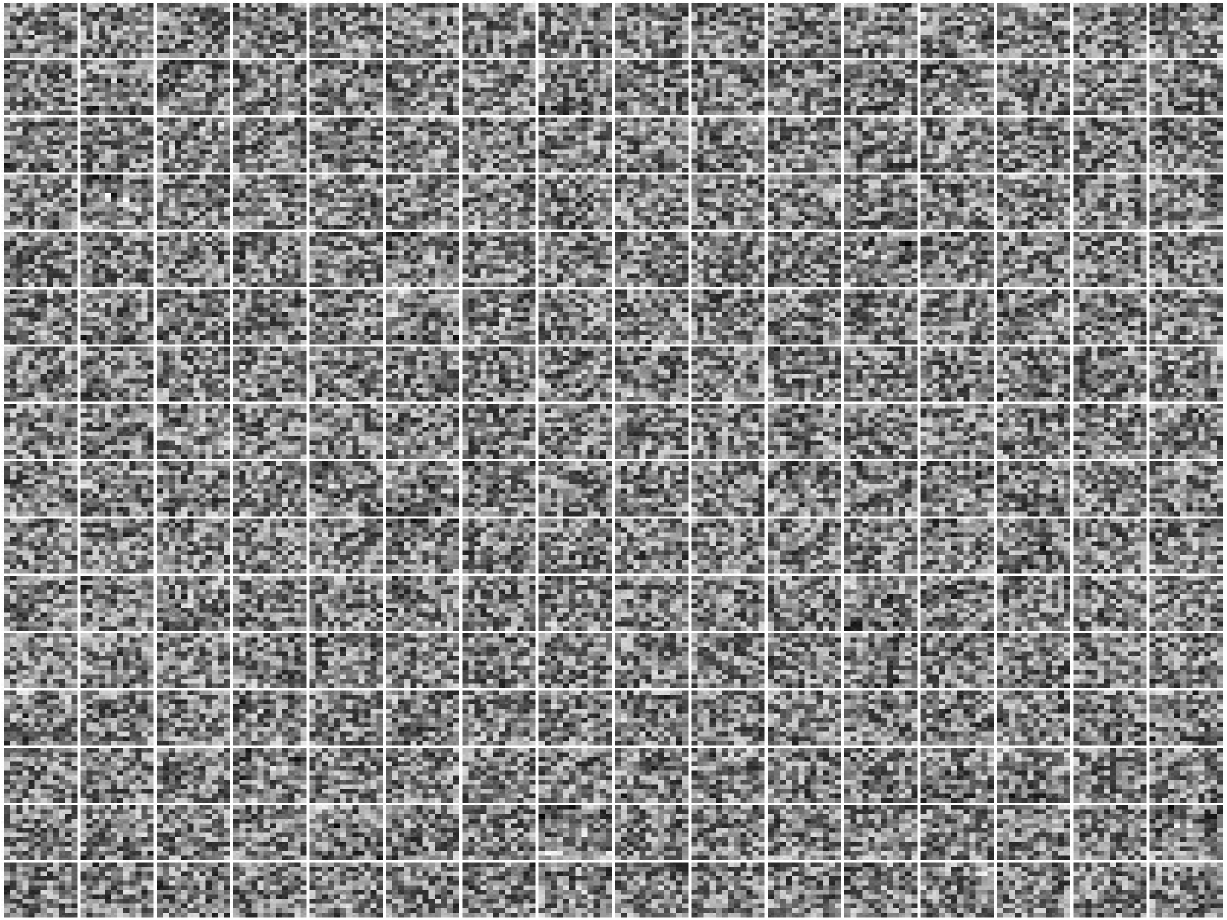
Sparse AE on handwritten digits (MNIST)

- **256 basis functionsBasis functions (columns of decoder matrix) are digit parts**
- **All digits are a linear combination of a small number of these**



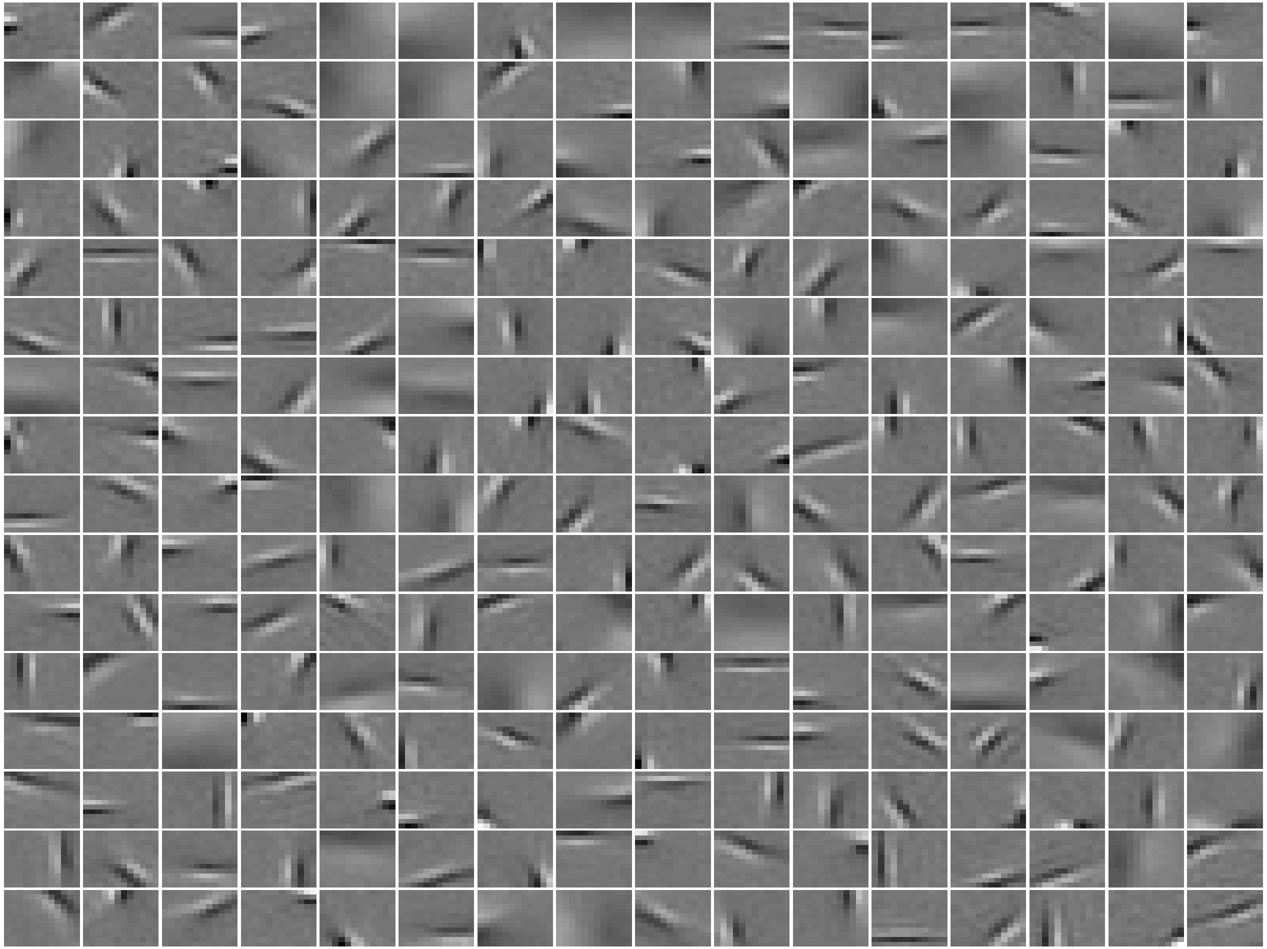
Predictive Sparse Decomposition (PSD): Training

- **Training on natural images patches.**
  - 12X12
  - 256 basis functions
  - [Ranzato 2007]



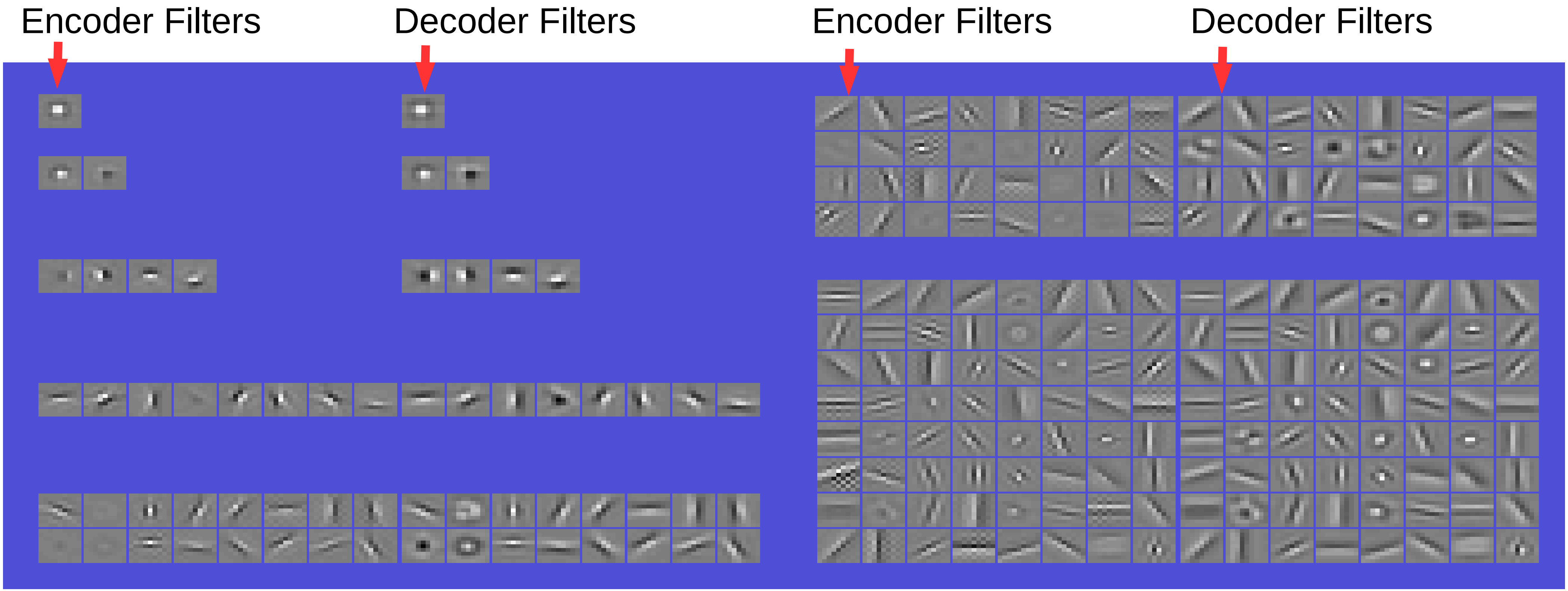
iteration no 0

Learned Features: V1-like receptive fields



Convolutional Sparse Auto-Encoder on Natural Images

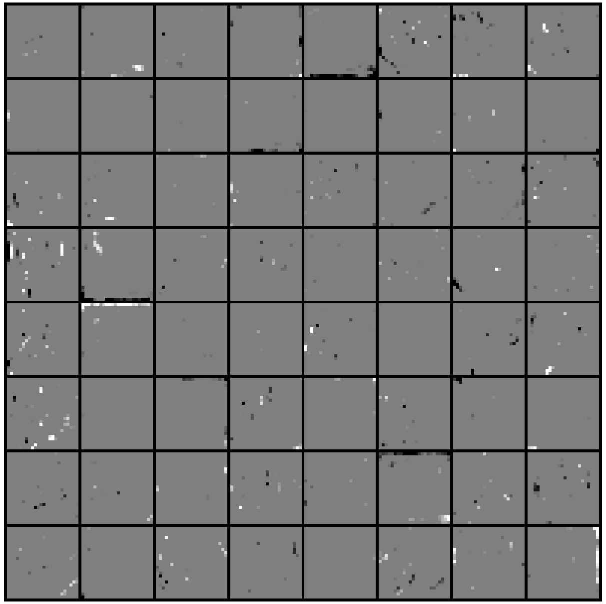
- **Filters and Basis Functions obtained. Linear decoder (conv)**
  - with 1, 2, 4, 8, 16, 32, and 64 filters [Kavukcuoglu NIPS 2010]



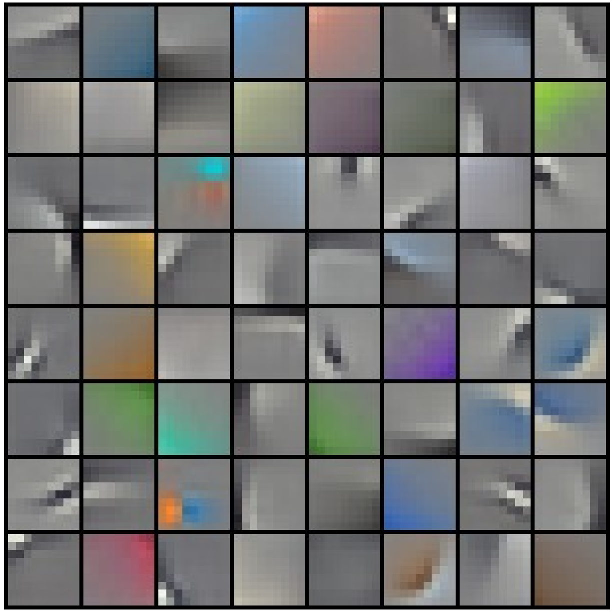
Convolutional Sparse Auto-Encoder on Natural Images

- Trained on CIFAR 10 (32x32 color images)
- Architecture: Linear decoder, LISTA recurrent encoder
- Pytorch implementation (talk to Jure Zbontar)

sparse codes (z) from encoder

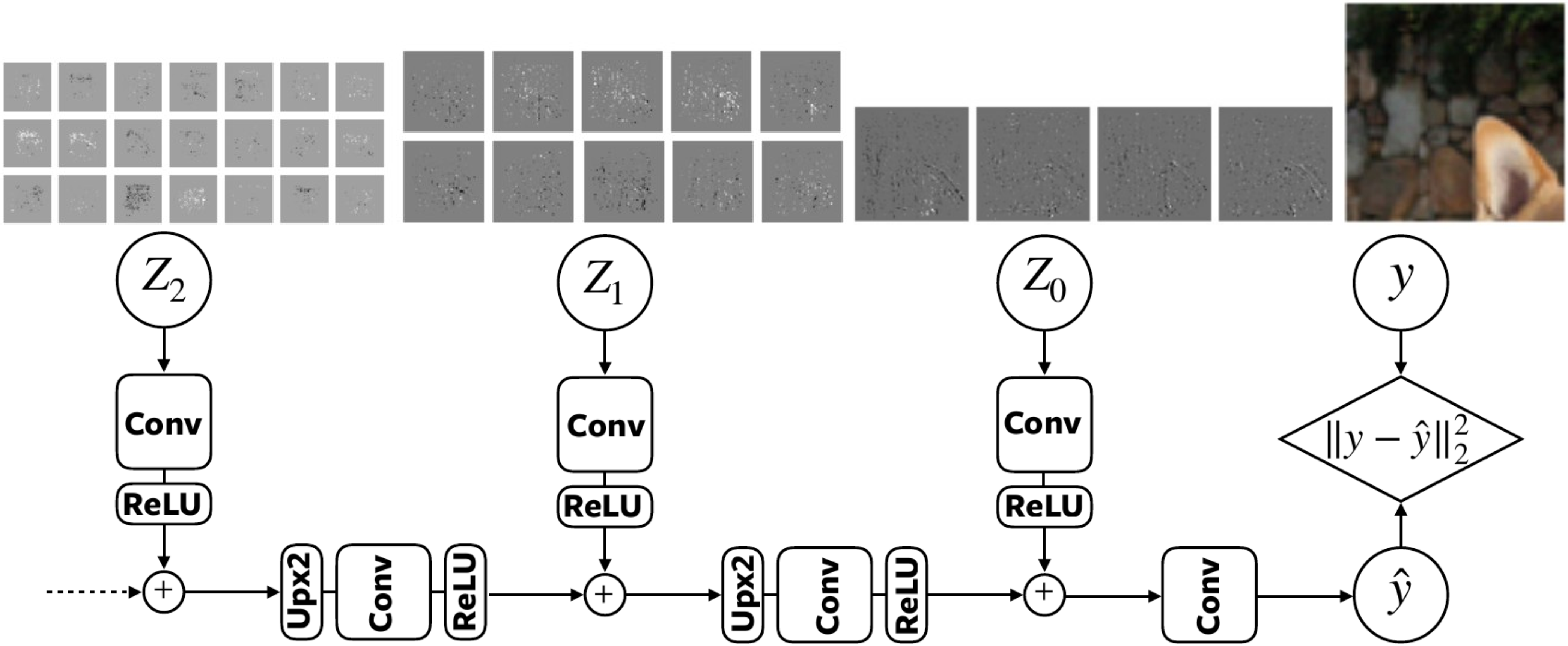


9x9 decoder kernels



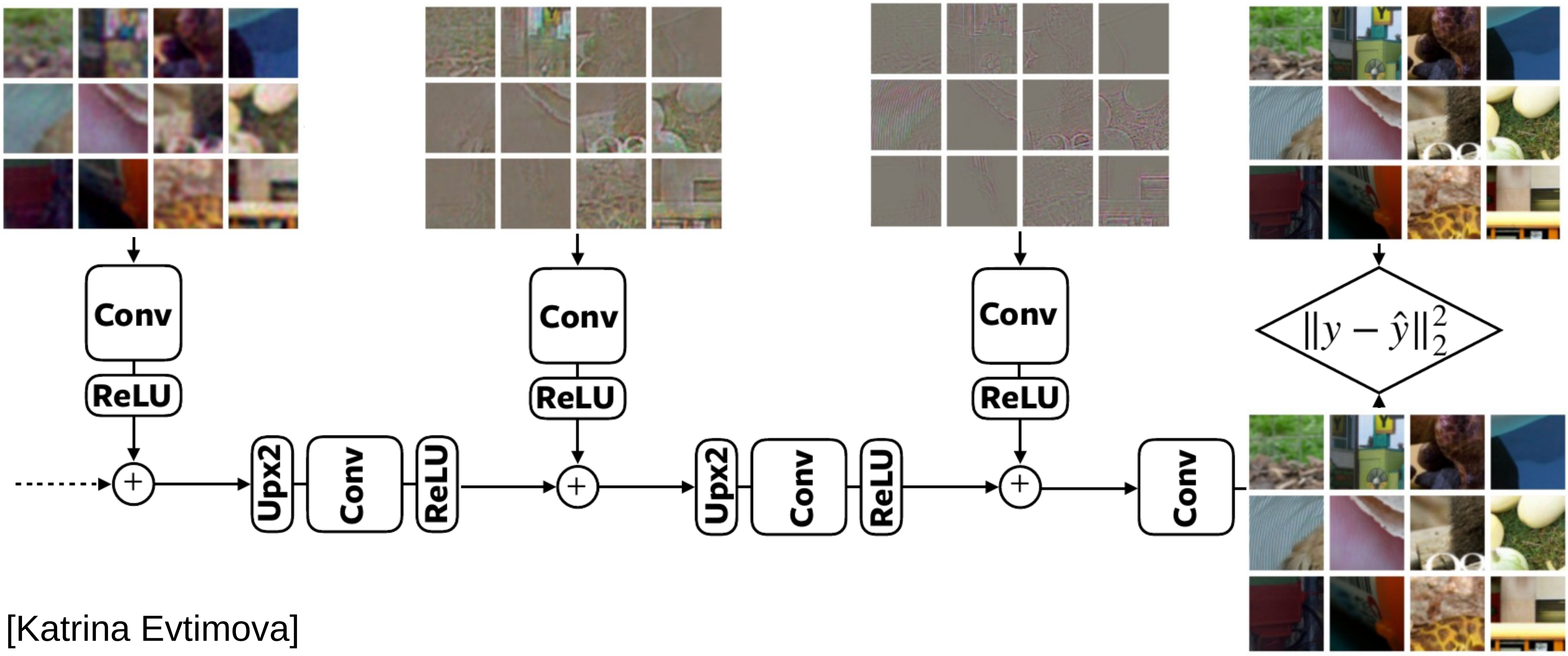
Multilayer Convolutional Sparse Modeling

- **Learning hierarchical representations**



Multilayer Convolutional Sparse Modeling

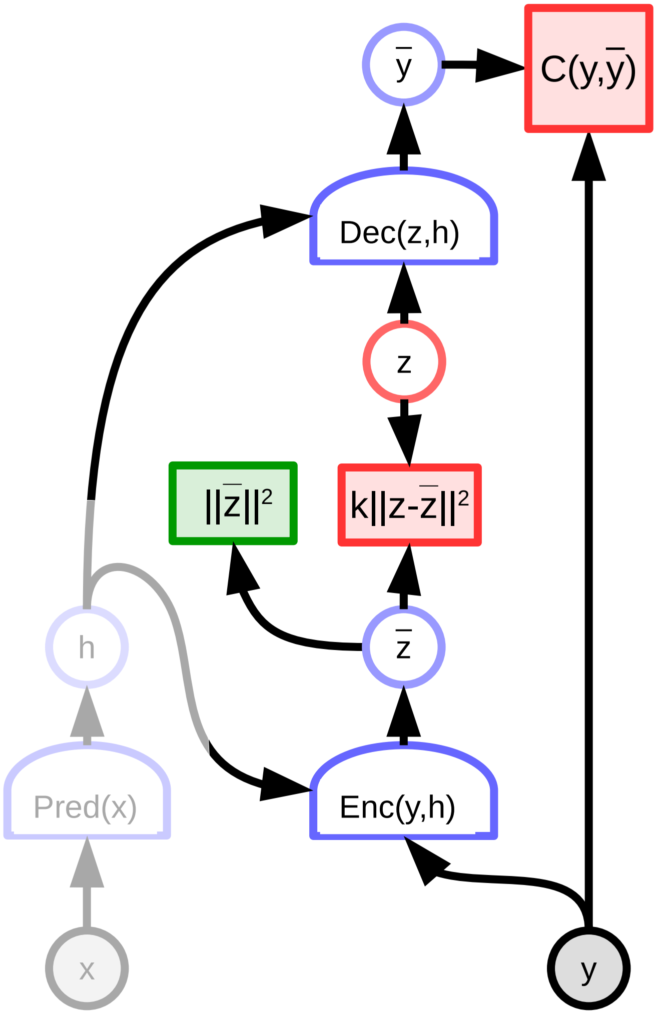
- Reconstructions from Z2, Z1, Z0 and all of (Z2,Z1,Z0)



[Katrina Evtimova]

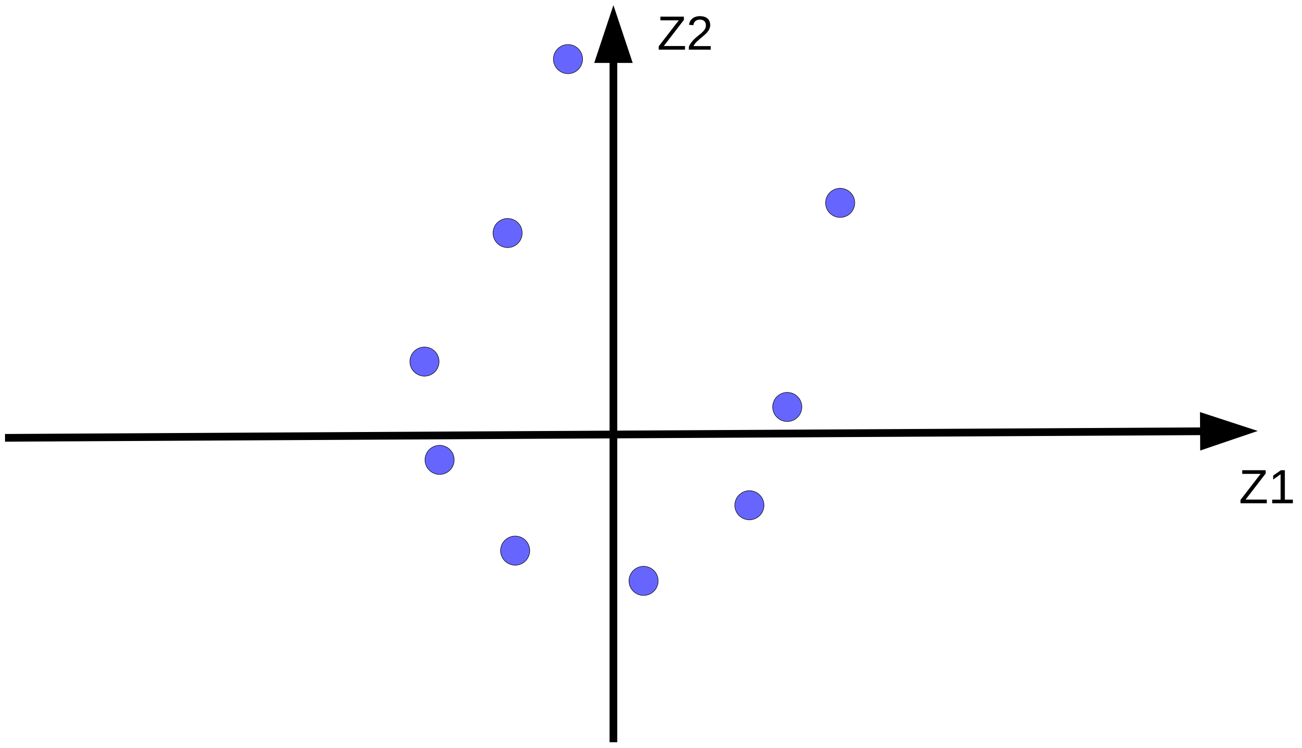
Variational Auto-Encoder

- **Limiting the information capacity of the code by adding Gaussian noise**
- **The energy term $k||z-\bar{z}||^2$ is seen as the log of a prior from which to sample z**
- **The encoder output is regularized to have a mean and a variance close to zero.**



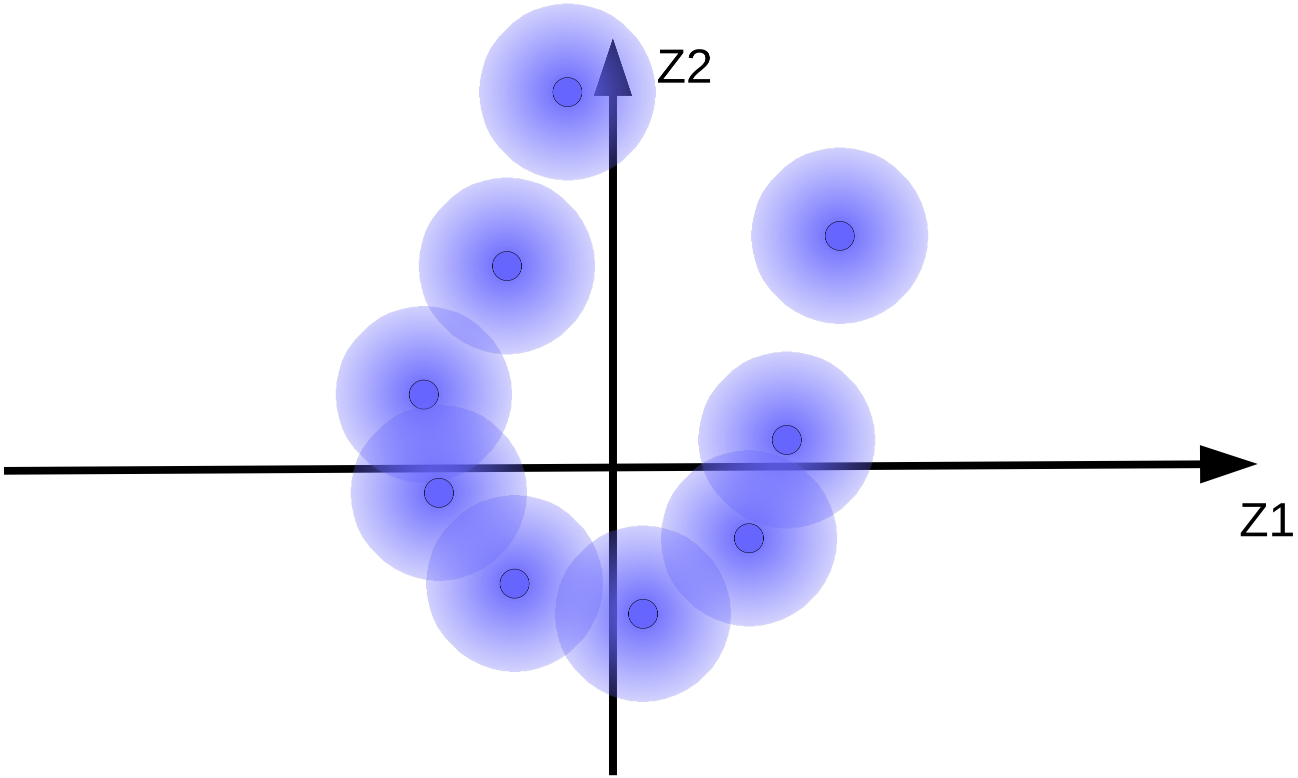
Variational Auto-Encoder

- **Code vectors for training samples**



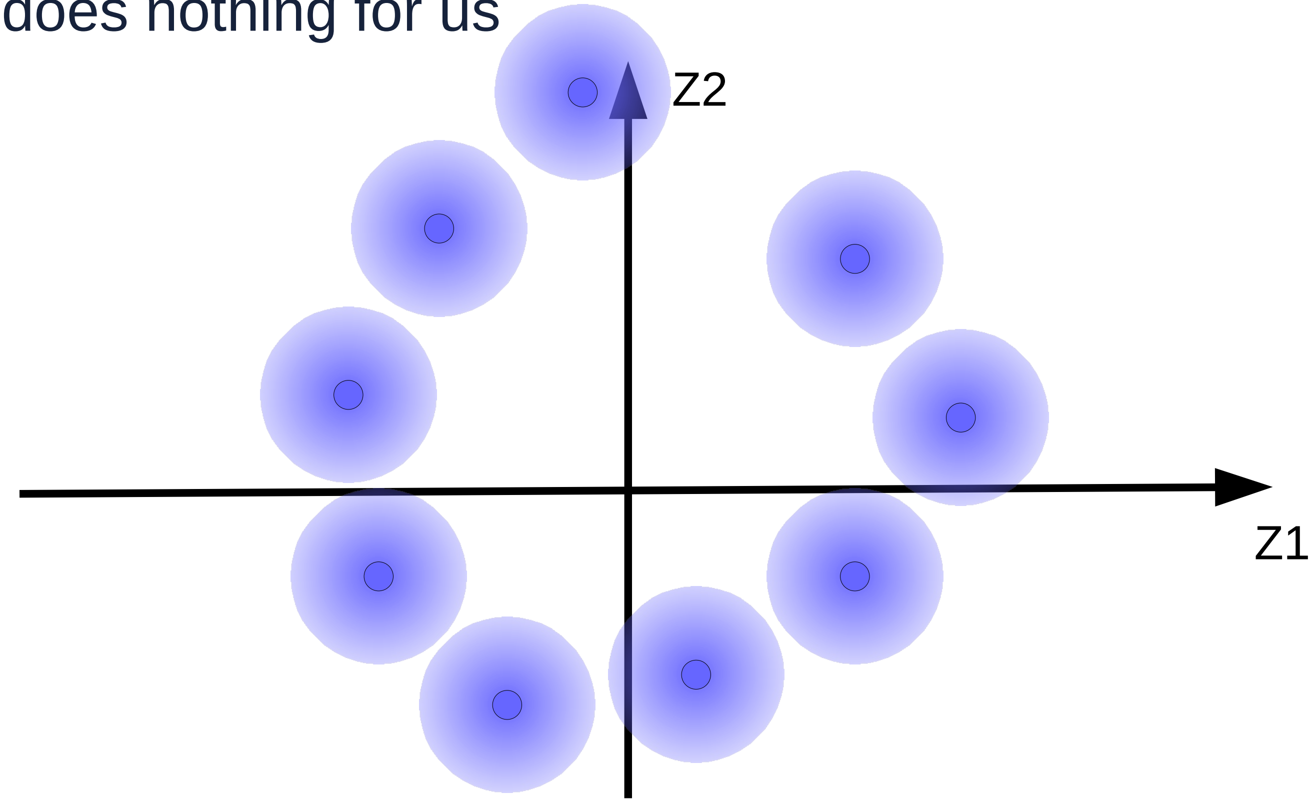
Variational Auto-Encoder

- **Code vectors for training sample with Gaussian noise**
  - Some fuzzy balls overlap, causing bad reconstructions



Variational Auto-Encoder

- **The code vectors want to move away from each other to minimize reconstruction error**
  - But that does nothing for us



Variational Auto-Encoder

- **Attach the balls to the center with a sping, so they don’t fly away**
  - Minimize the square distances of the balls to the origin
- **Center the balls around the origin**
  - Make the center of mass zero
- **Make the sizes of the balls close to 1 in each dimension**
  - Through a so-called KL term